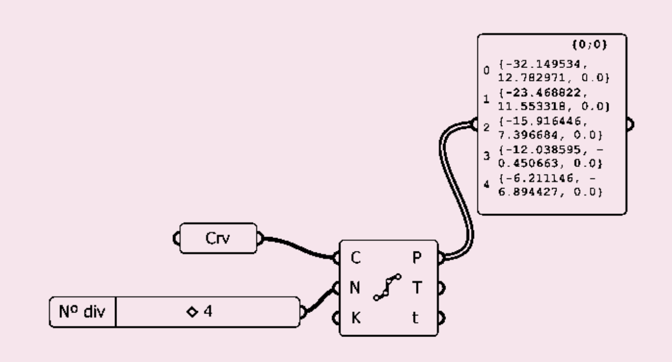
MODELADO PARAMÉTRICO CON GRASSHOPPER. INTRODUCCIÓN.

por

Mª ISABEL GÓMEZ SÁNCHEZ.



CUADERNOS

DEL INSTITUTO

JUAN DE HERRERA

DE LA *ESCUELA DE ARQUITECTURA* DE *MADRID*

5-68-02

MODELADO PARAMÉTRICO CON GRASSHOPPER. INTRODUCCIÓN.

por

Mª ISABEL GÓMEZ SÁNCHEZ

CUADERNOS

DEL INSTITUTO

JUAN DE HERRERA

DE LA *ESCUELA DE*

ARQUITECTURA

DE MADRID

5-68-02

C U A D E R N O S
D E L I N S T I T U T O
J U A N D E H E R R E R A

NUMERACIÓN

5 Área
68 Autor
02 Ordinal de cuaderno (del autor)

TEMAS

1 ESTRUCTURAS
2 CONSTRUCCIÓN
3 FÍSICA Y MATEMÁTICAS
4 TEORÍA
5 GEOMETRÍA Y DIBUJO
6 PROYECTOS
7 URBANISMO
8 RESTAURACIÓN
0 VARIOS

Modelado paramétrico con Grasshopper. Introducción.

Instituto Juan de Herrera.
Escuela Técnica Superior de Arquitectura de Madrid.
Gestión y portada: Ana Moure Rosende.
CUADERNO 462.01 / 5-68-02
ISBN-13: 978-84-9728-556-8
Depósito Legal: M-35368-2016

Grasshopper

Grasshopper es un plug-in para Rhinoceros, con el que podemos construir modelos utilizando todas las funciones que ofrece este potente programa de modelado 3D.
Utiliza un lenguaje de programación visual, en el que se crean programas arrastrando componentes en el área de trabajo del plug-in. Los componentes tienen entradas y salidas, que se conectan mediante cables a los parámetros requeridos o a otros componentes (sin más que arrastrar el ratón de unos a otros), según requerimientos, para definir las operaciones que realizarán nuestros programas. Éstos también pueden contener algoritmos numéricos y de texto, así como definiciones y relaciones lógicas.

Grasshopper permite el diseño paramétrico de modelos, al trabajar con entidades cuyos datos de entrada y salida pueden definirse de forma variable.

En este cuaderno vamos a presentar con ejemplos la forma de trabajo de Grasshopper, que tiene algunas particularidades pero permite aprovechar todas las capacidades de Rhino y, al igual que éste, ofrece múltiples maneras de construir nuestros modelos y obtener los resultados buscados.

¿Cómo se abre el plug-in?

Grasshopper fue desarrollado por David Rutten en Robert McNell & Associates, pero su uso se ha extendido a través de Internet y existen infinidad de recursos, definiciones de elementos, ejemplos de aplicación y modelos resueltos, compartidos en la red.

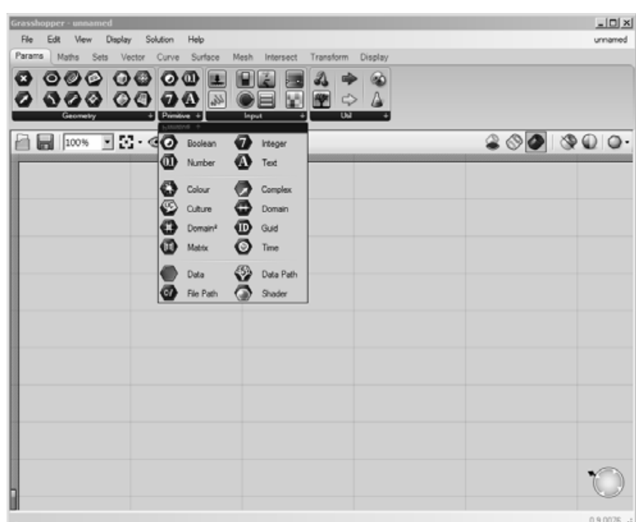
El plug-in (que como comentamos es en realidad un complejo programa que incluye todas las opciones, órdenes y entidades de Rhino), se puede descargar desde la página oficial de Grasshopper:

http://www.grasshopper3d.com

En ella encontraréis no sólo el programa (actualmente en versiones para Rhino 4.0 y 5.0, ampliamente desarrolladas desde hace años para Windows, así como una para Mac), sino también tutoriales, ejemplos, artículos, foros de discusión y multitud de recursos y aportaciones, tanto del propio Rutten y de otros colaboradores de McNell, como de la amplia comunidad de usuarios que ha creado.

Una vez instalado en nuestro ordenador, se ejecuta sin más que teclear "Grasshopper" en la barra de comandos de Rhino.

El aspecto que presenta es el de esta imagen:



> Existen múltiples video-tutoriales y manuales en Internet. Fundamental resulta el de Andrew Payne y Rajaa Issa, desarrollador de McNeel, que está traducido al español en este enlace:
>
> https://sistemasderepresentacion2.files.wordpress.com/2010/03/manual-grasshopper_espanol.pdf

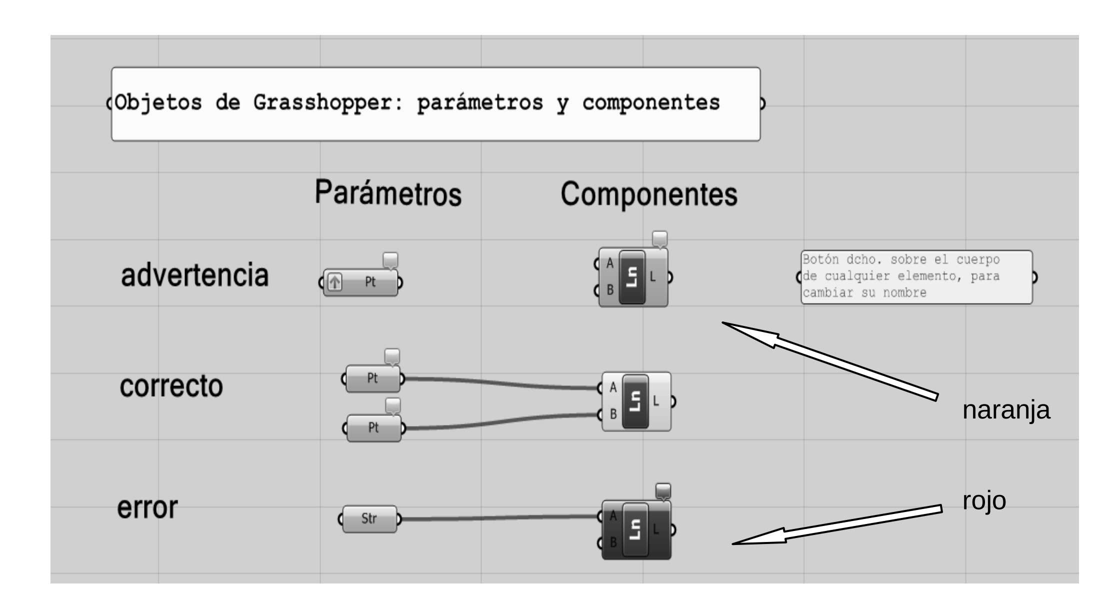
Como vemos, ofrece las barras de herramientas Params / Maths / Sets / Vector / Curve / Surface / Mesh / Intersect / Transform / Display, cada una de las cuales a su vez contiene otros submenús con distintos componentes.

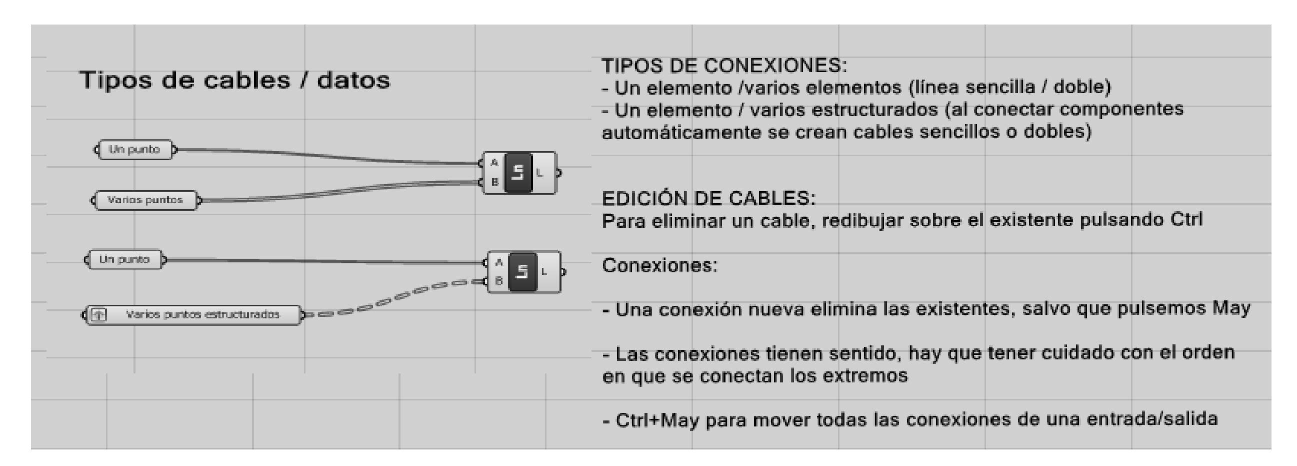
Para colocar un componente en el lienzo de trabajo, no tenemos más que hacer clic sobre él en el menú donde se encuentre, arrastrarlo hasta el lienzo y dejarlo en éste volviendo a hacer clic en el punto deseado.

Interfaz. Parámetros y componentes

Nuestro objetivo no es ofrecer un manual del programa, puesto que, como hemos comentado, ya hay varios y muy completos disponibles en Internet, sino aportar indicaciones de trabajo lo más útiles posible. Para ello nos vamos a permitir incluir observaciones en distintos formatos.

Comenzaremos por lo más simple: los tipos de elementos de trabajo y sus conexiones.
A los primeros los denominaremos COMPONENTES (en el ejemplo de la imagen, una línea, "Ln").
Los componentes tienen datos de entrada (PARÁMETROS) y de salida, que deben ser del tipo adecuado; si son correctos se mostrarán en color gris.
En caso contrario, si no hay error sino sólo advertencia (en la imagen, por ejemplo que no estén seleccionados los puntos que definen la línea), éstos aparecerán en naranja.
En caso de error (en nuestro ejemplo, que el dato de entrada sea una superficie en lugar de un punto), el componente línea aparecerá destacado en rojo.





Tipos de conexiones

Los cables pueden conectar elementos de uno en uno (línea sencilla) o múltiples (línea doble); así como listas de elementos ordenados según una determinada ley -datos estructurados-, en cuyo caso la línea doble será discontinua.

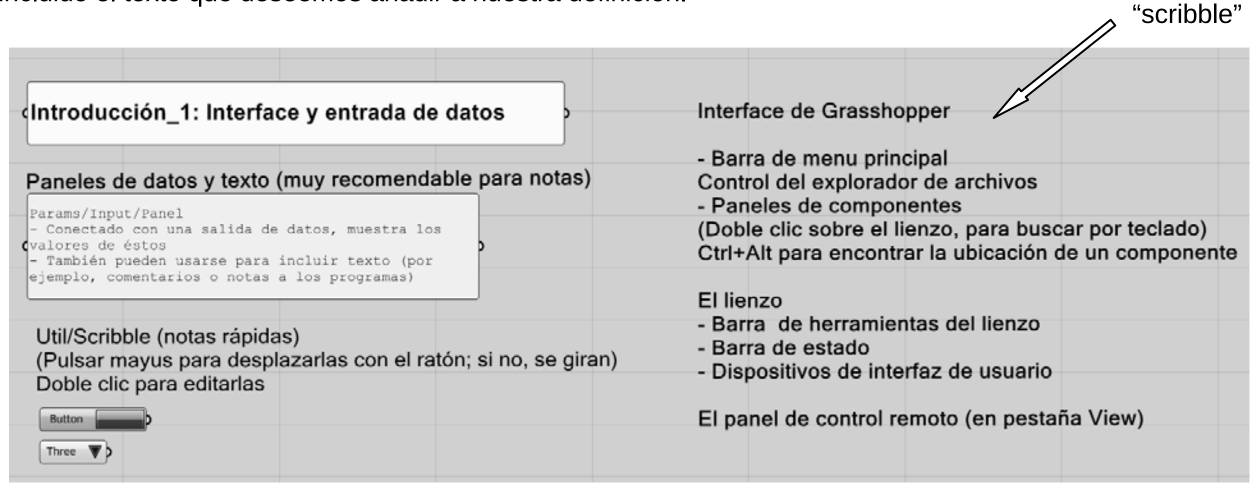
Los cables de uno u otro tipo se crean automáticamente en función de los datos que conecten, por lo que no necesitamos hacer nada para escogerlos. Pero sí podemos moverlos, crear nuevos, añadir a un mismo enlace, etc. En la imagen anterior se incluyen algunos consejos para el manejo de conexiones.

Anotaciones y paneles de datos

Podemos añadir comentarios a nuestros programas, lo cual resulta muy útil para explicar qué estamos haciendo en la definición de las rutinas que vayamos creando.

- Podemos hacerlo incluyendo notas rápidas (Params/Util/Scribble)
- O bien utilizar paneles de datos (Params/Input/Panel), que pueden contener cualquier tipo de datos, incluido el texto que deseemos añadir a nuestra definición.

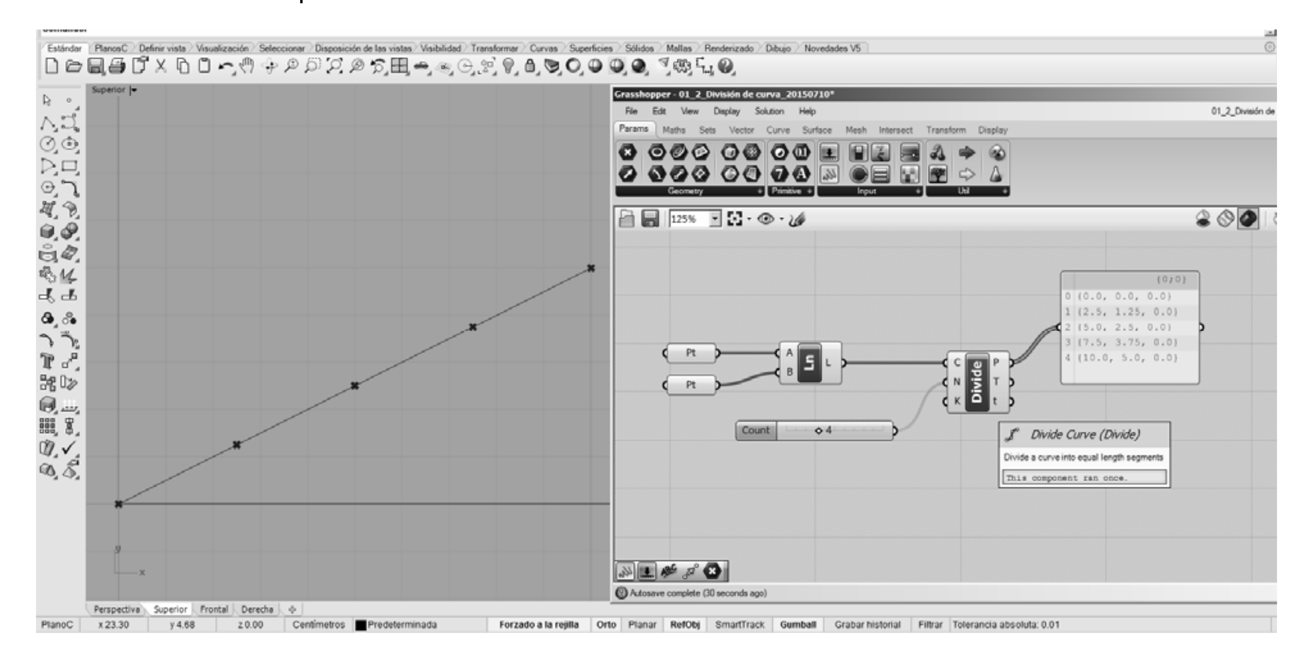
"scribble"



En ambos casos, como sucede con cualquier elemento que incluyamos en nuestras definiciones, todo es editable. Fuente, tipo y tamaño de letras, interlineado y justificación del texto, color de fuente, de fondo del panel, etc. Se pueden añadir numeraciones y viñetas y disponemos de todas las opciones de un editor de textos.

Conectado a un componente, un panel de datos mostrará la salida de datos correspondiente.

Veámoslo en nuestra primera definición:



En ella hemos creado una línea definida por dos puntos (uno lo hemos situado en (0,0,0) y el otro en (10,5,0), y la hemos dividido en un número variable de partes iguales (en el momento de la imagen, 4).

Necesitaremos los siguientes componentes:

- Punto ("Pt"). Para crear dos, podemos arrastrarlo dos veces al lienzo, o tecleando Ctrl+c/Ctrl+v;
- Línea ("Ln", en Curve/primitive);
- Divide ("Divide", en Curve/Division).

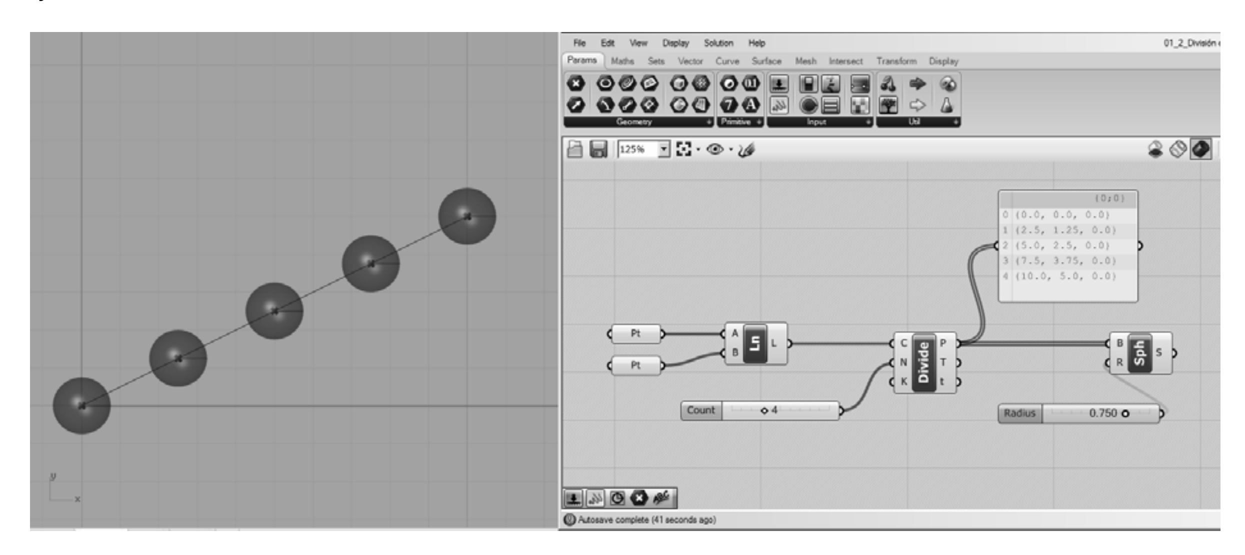
Nos falta tan sólo un **deslizador** ("**Number slider**", en Params/Input), verde en nuestra imagen, que conectamos a la entrada N (nº de divisiones) de "Divide".

Con botón dcho. del ratón sobre el deslizador podemos definir el tipo de elemento (entero, ya que es nº de divisiones), los límites mínimo y máximo (por ejemplo 1 y 10) y el valor actual (lo dejamos en 4).

El panel de datos que hemos conectado a la salida de "Divide" nos muestra las coordenadas de los puntos en que hemos dividido el segmento (inicial, final y los tres intermedios).

División de curva

Colocamos una esfera ("Sphere", en Surface/Primitive) en cada uno de los puntos de división, y le damos valor al radio (entrada R de "Sph") con un deslizador de tipo "floating", cuyo valor inicialmente fijamos en 0.75.



A continuación vamos a dividir la línea no en un nº determinado de partes iguales, sino en un nº determinado de segmentos de longitud dada.

Para ello necesitamos utilizar un nuevo componente; "**Divide Length**" (en Curve/Division).

Damos, con un deslizador tipo "floating", el valor 4.5 al parámetro L (longitud de las divisiones) del componente "Divide Length", y observamos que sólo caben tres esferas, y sobra parte del segmento (ver figura siguiente).

Inserción de un collar de esferas tangentes entre sí en una curva

Para que las esferas sean tangentes, basta con que su radio mida la mitad de la distancia entre dos puntos de la subdivisión.

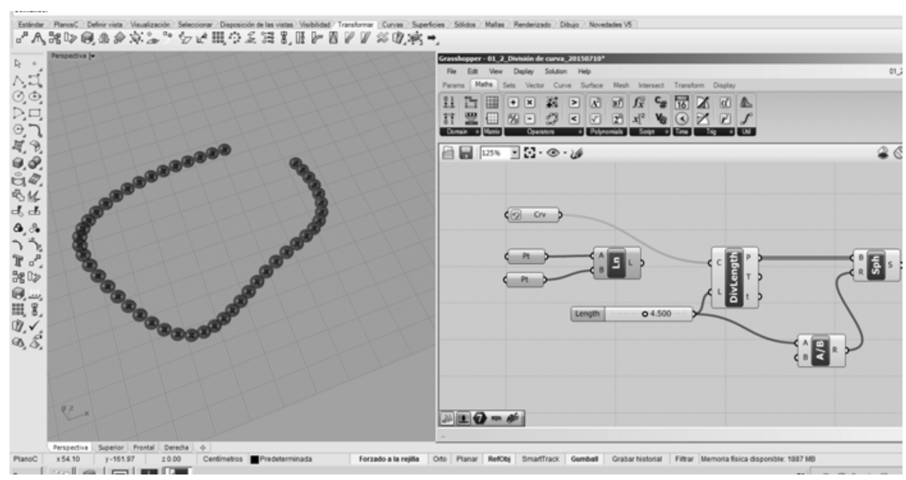
Utilizando el componente "Division" (en Maths/Operators), dividimos L entre 2, sin más que conectar el deslizador "L" de "Divide Length" a la entrada "A" y el valor 2 a la entrada "B" de "A/B".

Esto segundo puede hacerse con otro deslizador o simplemente haciendo clic con el botón derecho sobre la entrada B y tecleando para "Set data item" el valor 2.

Esta misma definición puede aplicarse a cualquier curva, sin más que seleccionar en el parámetro C (curva a dividir) de "Divide Length" la curva escogida, en lugar del segmento definido por dos puntos con el que hemos estado trabajando.

(Insertamos un componente "Crv" (en Params/Geometry) y "reconectamos" el parámetro de entrada de datos de "Divide Length".

Btn. dcho. sobre "Crv/Set one curve" y seleccionamos una curva que previamente hayamos dibujado en Rhino).



Estructuras de datos. Series, rangos e intervalos

El manejo de datos en Grasshopper es de vital importancia. Con frecuencia se realizan operaciones no con datos aislados, sino con series o familias de elementos que han de estar bien definidos para que las operaciones sean correctas.

Comenzaremos repasando conceptos básicos. Para ello hemos definido:

Serie

- Una “Serie” de 10 elementos con paso 5 y comienzo en 0.0;

Dominio

- Un “Dominio” o conjunto de valores comprendidos entre dos límites (en este caso, entre 0.0 y 10.0);

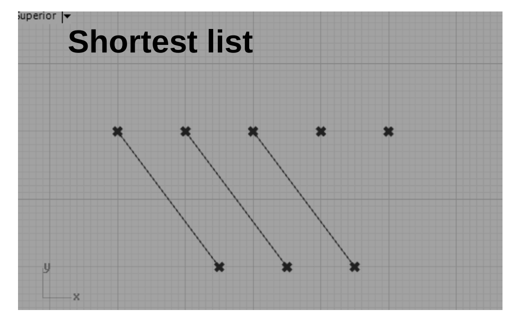
Rango

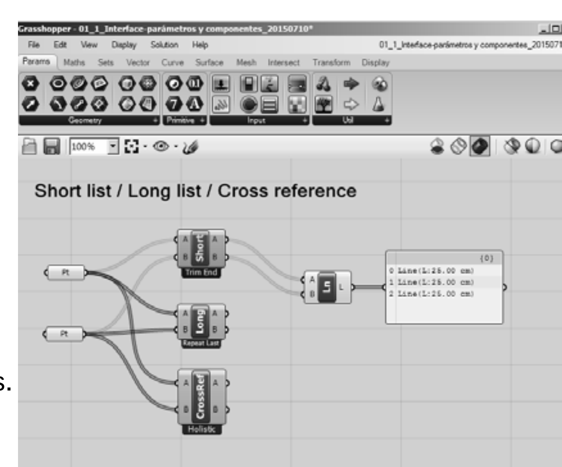
- Un “Rango” de N “saltos” o intervalos (hemos supuesto sea N=8) distribuidos entre los extremos del dominio anterior. 0.0, 1.25, 2.5, 3.75, 5.0, 6.25, 7.5, 8.75 y 10.0.

Nueve valores correspondientes a ocho intervalos.

Listas de datos: relaciones

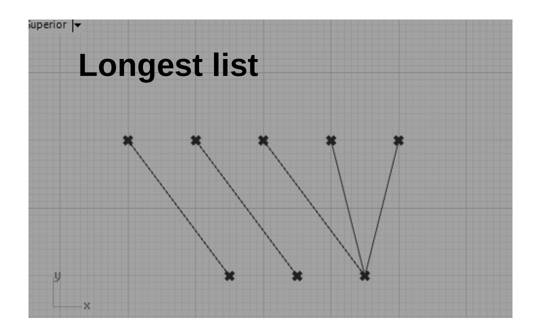
Otro aspecto que hay que tener en cuenta es la forma en que se relacionan los elementos de dos listas. Veámoslo con un ejemplo en el que la entrada de datos A corresponde a la fila superior de puntos de la imagen, y la entrada B corresponde a la fila inferior (en ambos casos capturados de izquierda a derecha)

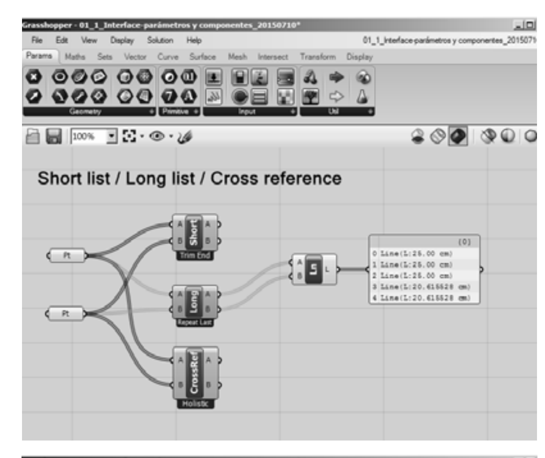




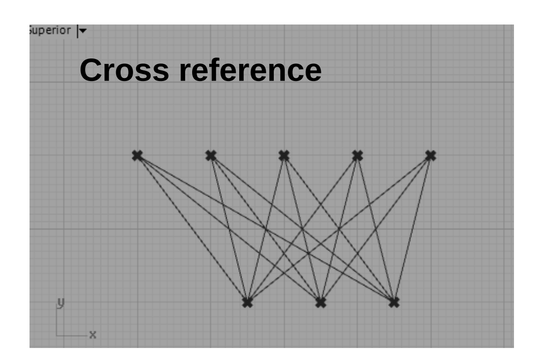
- “Shortest list”: une los datos que ocupan la misma posición en ambas listas, hasta que se agota una de ellas.

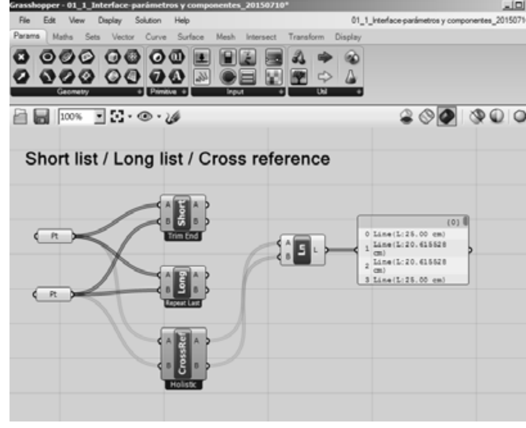
- “Longest list”: une el último elemento de la primera lista que se agota con los restantes de la lista más larga.





- “Cross reference” relaciona todos los datos entre sí.





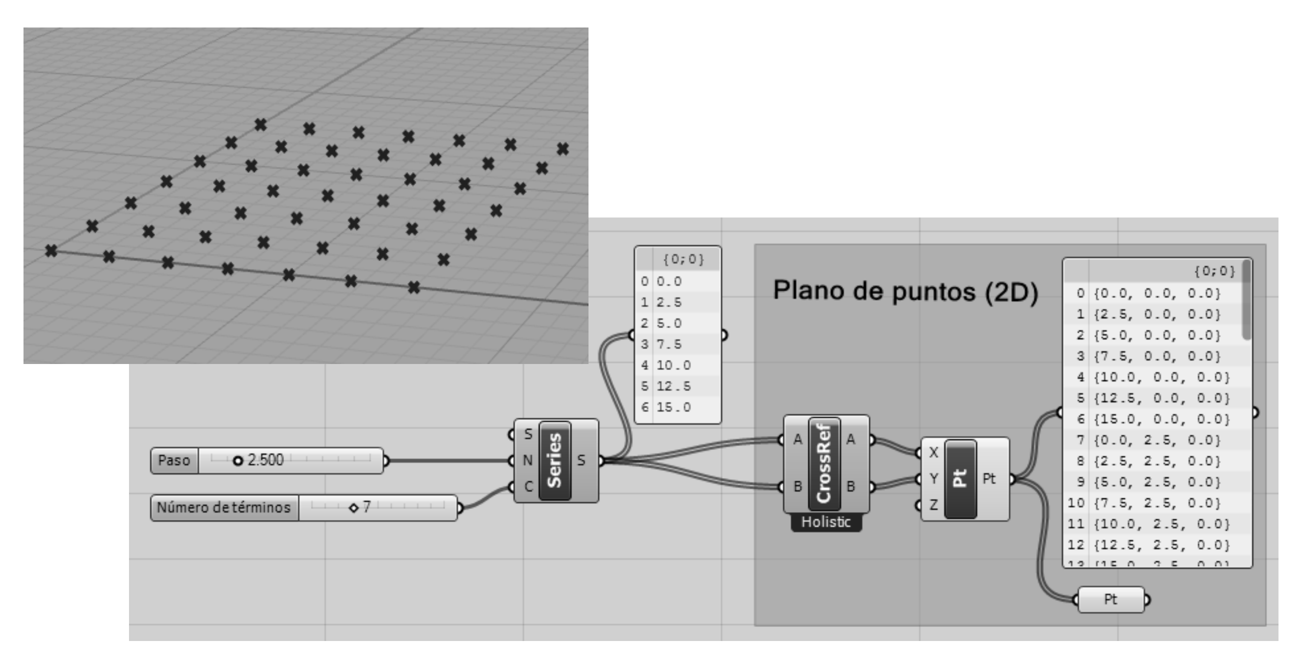
Todos ellos se encuentran en Sets/Lists.

Crear plano de puntos

Vamos a utilizar los conceptos que acabamos de ver para crear algunas definiciones geométricas sencillas. Comenzaremos creando un plano de puntos.

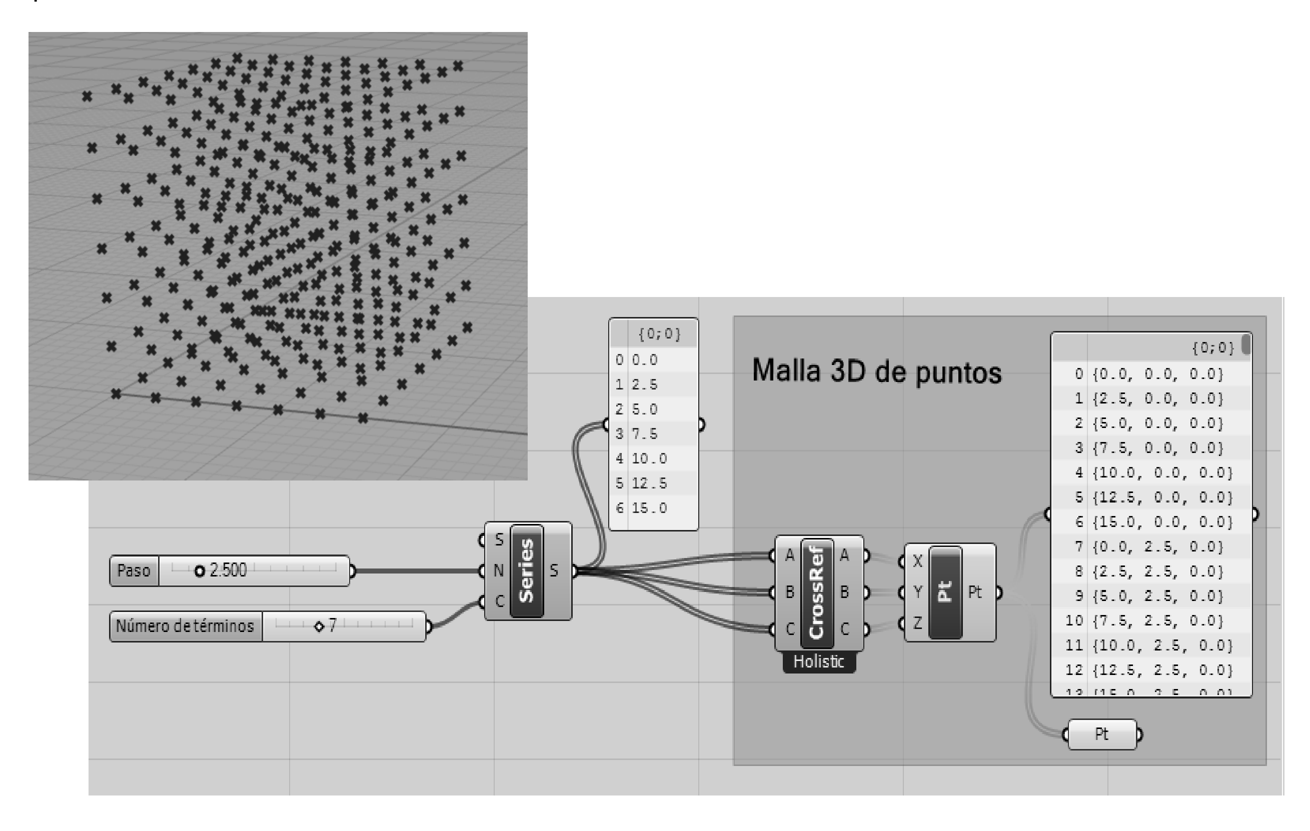
Lo hemos resuelto con una Serie de paso (N) = 2 y 6 elementos (C).

Basta con cruzar los valores de esta serie, conectándola a las dos entradas de “Cross Reference” para tener la red de puntos que estamos buscando. Sólo nos queda conectar un elemento “Punto” a cada una de las coordenadas para visualizarlos en Rhino.



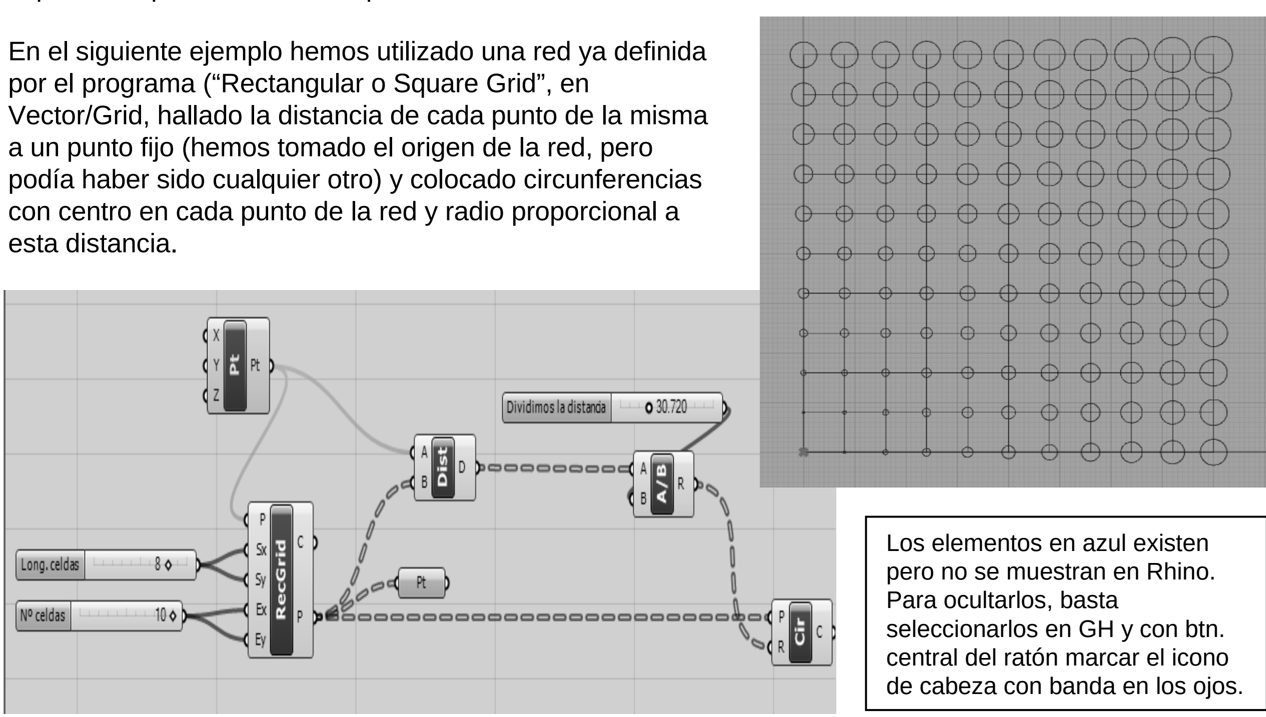
Crear matriz 3D de puntos

Aprovechando la definición anterior vamos a construir una red 3D de puntos. Sólo tenemos que utilizar un “Cross Reference” que cruce datos de tres variables, y asociarlos a las coordenadas X, Y, Z de nuestros puntos.

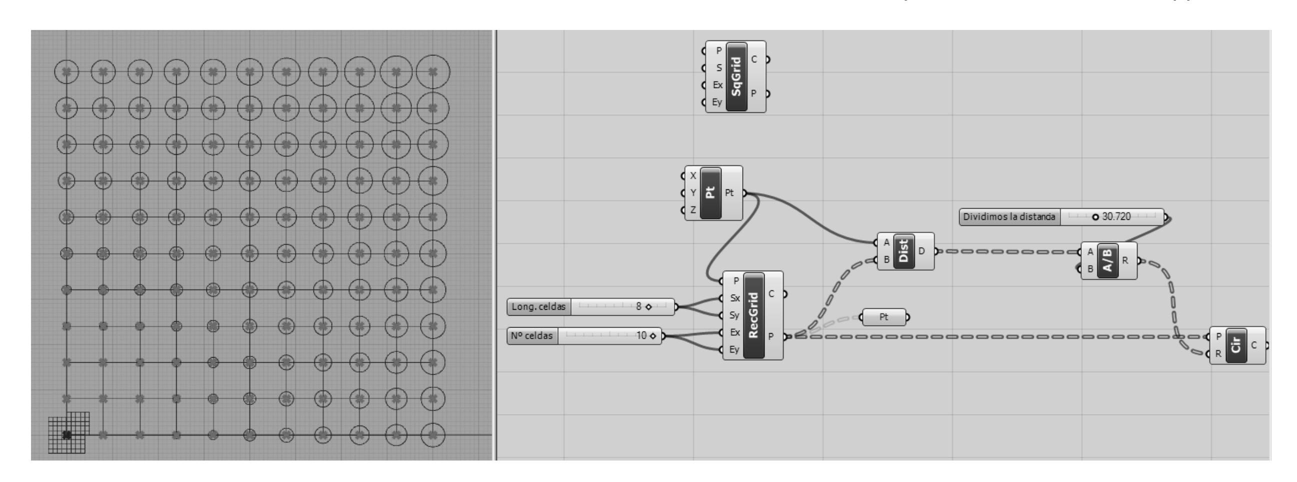


También podíamos haberlo resuelto definiendo dominios para cada una de las coordenadas de los puntos. Y podemos utilizar el componente “Surface From Points” (en Surface/Freeform) para construir una superficie a partir de la red de puntos.

En el siguiente ejemplo hemos utilizado una red ya definida por el programa (“Rectangular o Square Grid”, en Vector/Grid, hallado la distancia de cada punto de la misma a un punto fijo (hemos tomado el origen de la red, pero podía haber sido cualquier otro) y colocado circunferencias con centro en cada punto de la red y radio proporcional a esta distancia.



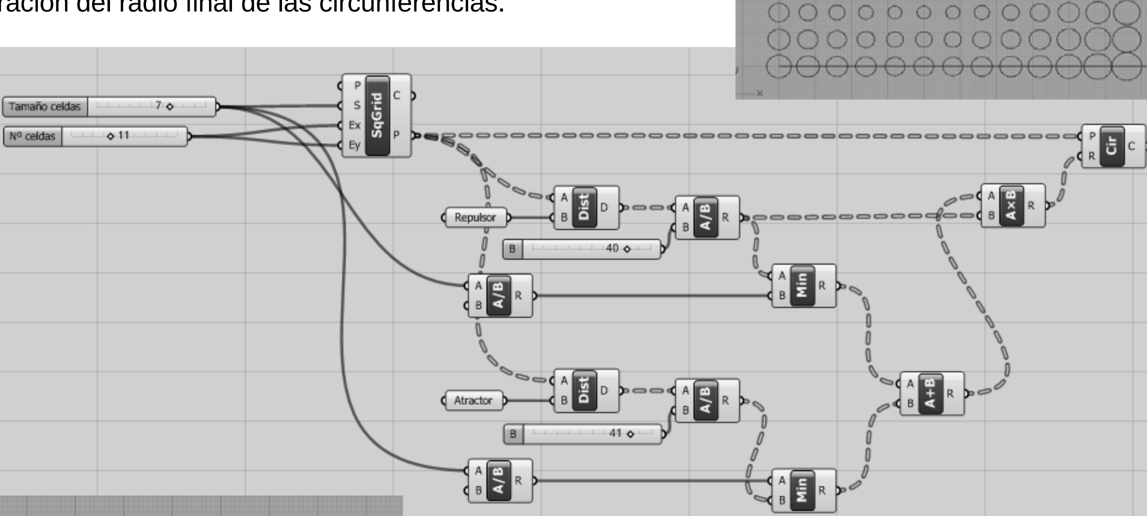
Los elementos en azul existen pero no se muestran en Rhino. Para ocultarlos, basta seleccionarlos en GH y con btn. central del ratón marcar el icono de cabeza con banda en los ojos.

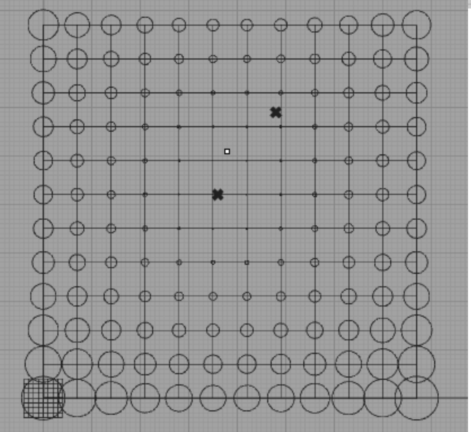


Véase la misma red, con puntos en sus nodos en este caso no ocultos sino visibles, y seleccionados (en GH todo lo seleccionado se muestra en verde).

Podemos utilizar otro punto distinto al origen para medir distancias (imagen dcha.) o trabajar con más de un punto atractor/repulsor.

- En caso de considerar varios puntos de referencia, el radio de las circunferencias se calculará calculando la media de las distancias a éstos, tal y como se muestra en la siguiente definición. En ella hemos hallado los valores mínimos entre la distancia minorada y el lado de las celdas de la rejilla, sumado los valores obtenidos, y utilizado el resultado como coeficiente de ponderación del radio final de las circunferencias.



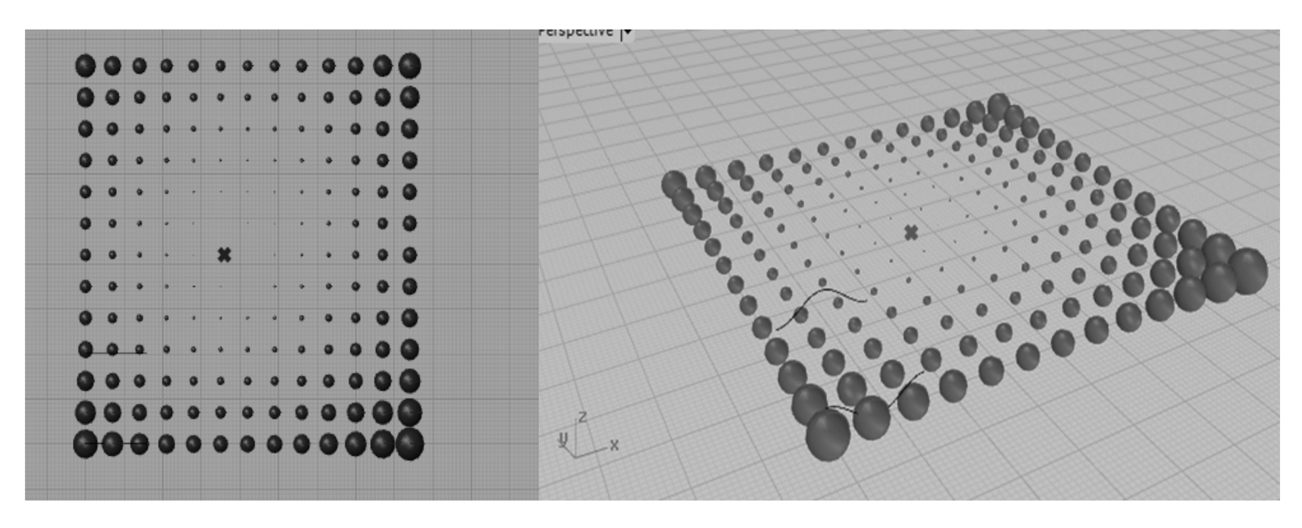


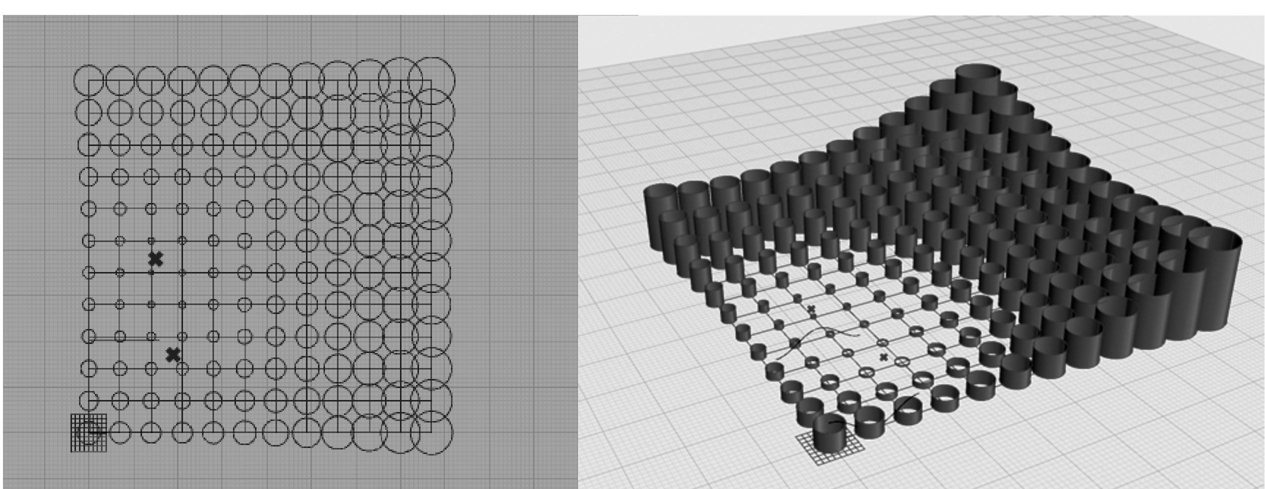
Este ejemplo puede utilizarse para colocar en los puntos de la malla elementos que nos permitan tener una imagen tridimensional de las distancias y el área de influencia de los puntos de referencia.

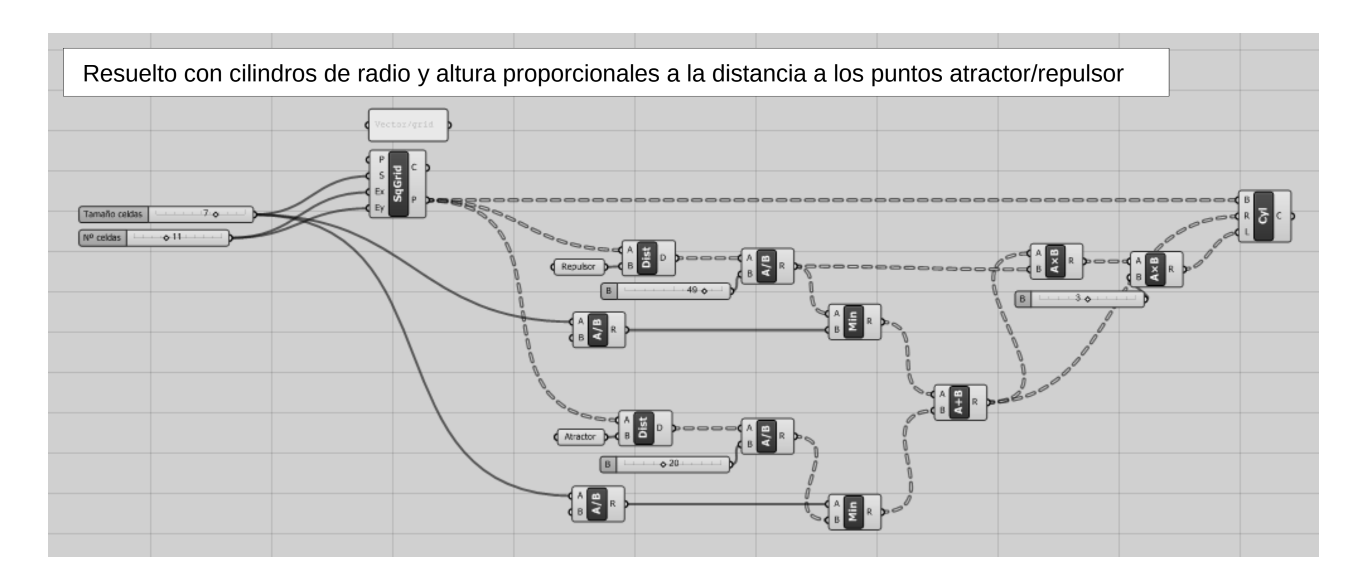
Es tan sencillo como colocar en cada punto de la red:

1. Esferas de radio proporcional al valor que hemos hallado para el ejemplo de las circunferencias (primera imagen de la página siguiente);

2. O cilindros de radio y altura proporcionales a este mismo valor (segunda imagen, de la que incluimos su definición, que coincide con la manejada en los ejemplos anteriores).







Resuelto con cilindros de radio y altura proporcionales a la distancia a los puntos atractor/repulsor

Los puntos serán atractores o repulsores dependiendo de que el tamaño de los elementos sea directa o inversamente proporcional a la distancia.
Las distancias se ajustarán, multiplicando o dividiéndolas por un factor variable, para que la dimensión que tendrán los elementos de la red sea proporcional al tamaño de ésta.

Red 3D de cubos

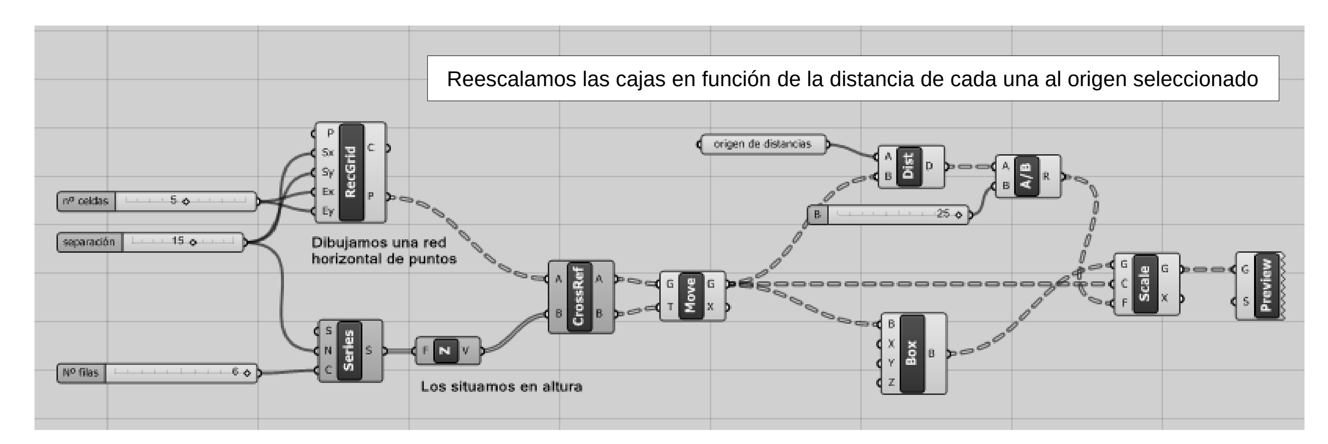
Y vamos a finalizar este ejemplo aplicándolo a la generación de una red tridimensional de cubos, en la que el tamaño de cada uno de ellos será proporcional a su distancia a un punto cualquiera escogido por nosotros.

Necesitaremos una matriz 3D de puntos en los que situar el centro de las cajas, para lo cual podemos repetir la definición del ejemplo anterior (utilizando “Cross Reference” en X, Y, Z; o bien generarla como vamos a explicar a continuación (para tener un nuevo método).

1. Empezamos dibujando una red XY de puntos;
2. Los situamos a la cota correspondiente desplazándolos (“Move”, en Transform/Euclidean) según un vector paralelo al eje Z (utilizando el componente “Unit Z”, en Vector/Vector);

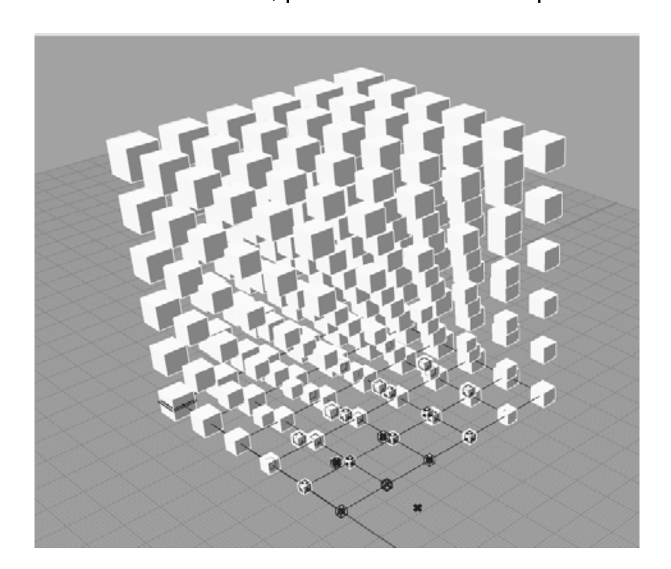
Como lo hacemos cruzando los datos de la serie utilizada para crear la red XY de la base, con “Cross Reference”, estaremos situando una columna sobre cada punto del plano XY.

3. Colocamos las cajas (“Domain Box”, en Surface/Primitive) en los puntos de la red 3D;
4. Las reescalamos (“Scale”, en Transform/Affine) proporcionalmente a su distancia (“Distance”, en Vector/Point) a un punto escogido.



El punto escogido como origen de distancias se ha situado en el plano de la base próximo al origen de la red (véase en la imagen), pero por supuesto podríamos haber escogido cualquier otra posición.

Hemos conectado un último componente, “Custom Preview”, en Display/Preview, para que se muestre en Rhino la red creada, pudiendo definir las opciones de visualización que estimemos oportunas.



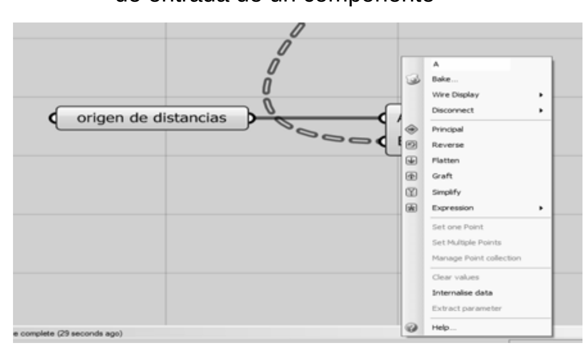
Pulsando el botón derecho del ratón sobre cualquier parámetro de un componente aparece un menú emergente que nos muestra todas las acciones y opciones de configuración disponibles para él.

Para determinar el valor de estas opciones tenemos distintas formas de entrada de datos. Veámoslo por ejemplo con el caso del componente “Distancia” (en la siguiente imagen):

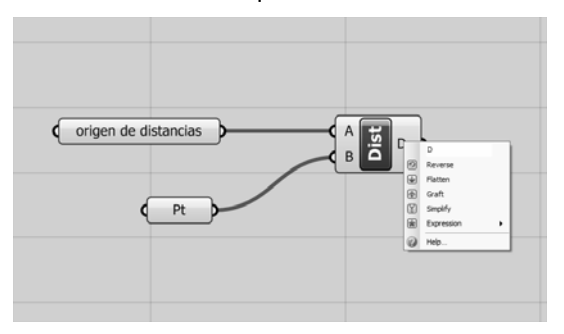
- Podemos conectar a los parámetros A y B los puntos correspondientes, creados como tales con el parámetro “Point”;

- Pero también podemos escoger la opción “Set one (o multiple) point” en el menú emergente, y seleccionar puntos ya creados en Rhino, o bien situarlos nosotros haciendo clic en el lugar deseado.

Panel de opciones de un parámetro de entrada de un componente

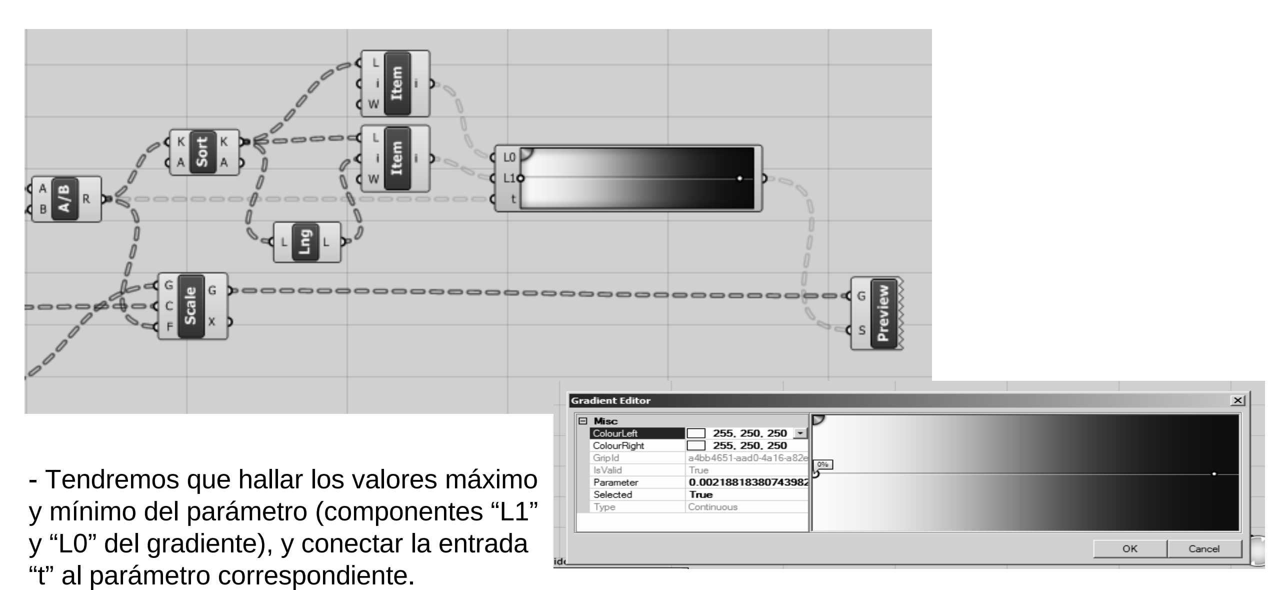


Panel de opciones del parámetro de salida del componente "Distancia"

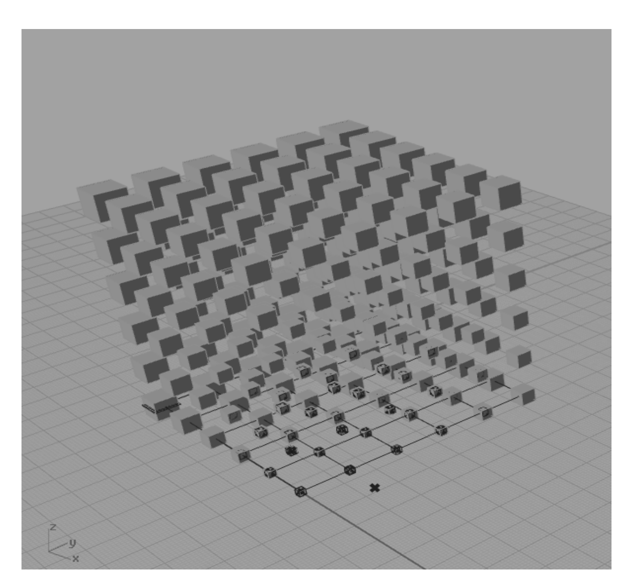


Y finalizaremos nuestra definición de red de cubos haciendo referencia a una herramienta muy útil para mostrar gráficamente la distribución de valores de cualquier parámetro (por ejemplo, en nuestro modelo, lo hemos asociado al valor de la distancia al punto de referencia, que es proporcional al tamaño de los cubos.

En vez de conectar directamente con "Preview" el componente "Scale", introducimos un **gradiente de colores** que tenga en cuenta los valores de estas distancias.



- Tendremos que hallar los valores máximo y mínimo del parámetro (componentes "L1" y "L0" del gradiente), y conectar la entrada "t" al parámetro correspondiente.



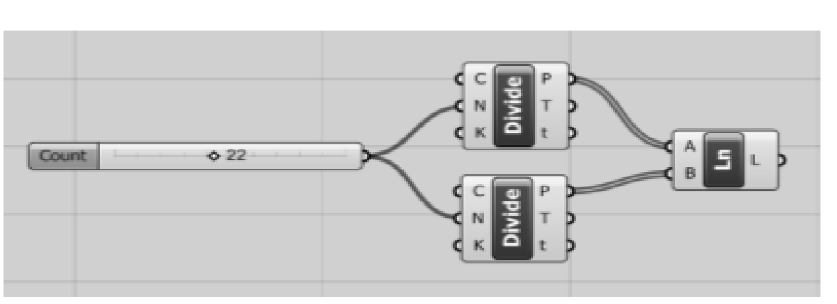
- A continuación configuraremos el gradiente, eligiendo los colores correspondientes a los rangos de valores que queramos definir (para mostrar el editor del gradiente basta hacer doble clic sobre este componente).

(Se pueden seleccionar tantos colores intermedios, en las franjas de valores que determinemos, como se desee).

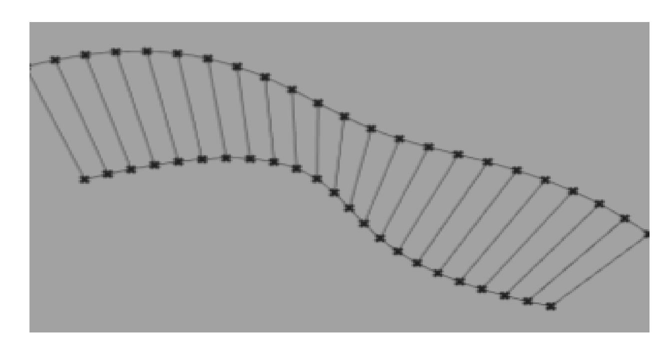
- Y finalmente conectaremos la salida del gradiente de colores al parámetro de entrada "Shader", *S*, de "Preview".

Curvas. División y cruce de datos

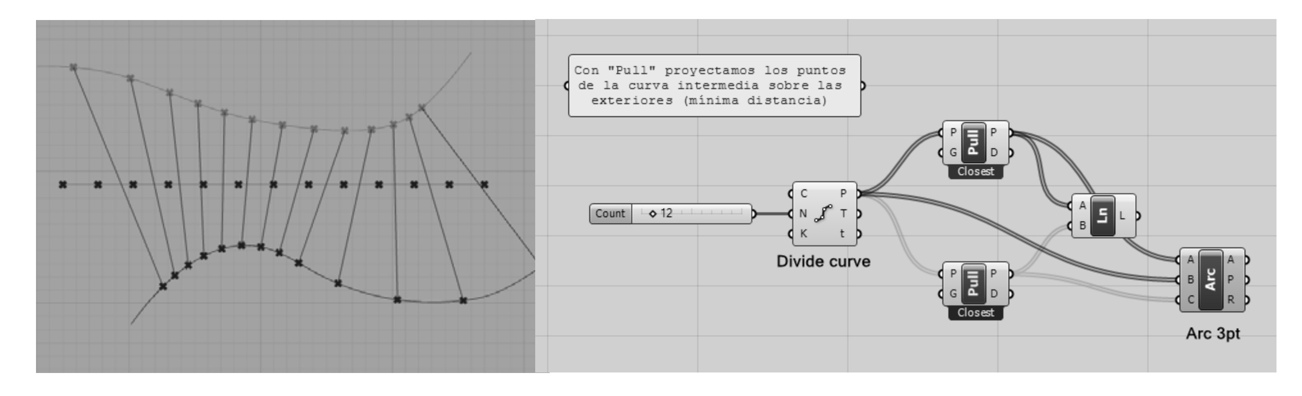
Como comentamos en la introducción, no es nuestro objetivo presentar un tutorial de GH, pues de hecho ya existen y muy buenos. Para quienes no lo hayan manejado nunca remitimos al “Grasshopper. Getting started by David Rutten”, disponible en Internet, que recoge en los 13 videos cuyos títulos incluimos a continuación los fundamentos de trabajo con GH. No os los perdáis.

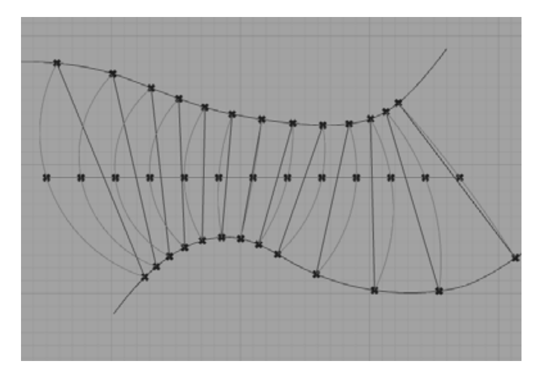


1. Interface basics
2. Multiple components
3. Refining a definition
4. Basic questions and answers
5. Creating biarcs
6. Displaying the definition with colors
7. Lofting multiple curves in GH
8. Creating components with custom default
9. How to get support and find samples
10. Using multiple inputs
11. Generic components
12. Lists and random
13. Shortcuts tips working with sliders



En relación con el trazado de curvas, se definen posibles formas de dividirlas y trazar líneas de conexión entre ellas. Empezaremos dividiendo dos curvas en un mismo número de partes, y uniendo los puntos mediante rectas (utilizando el componente “Divide Curve”, en Curve/Division).

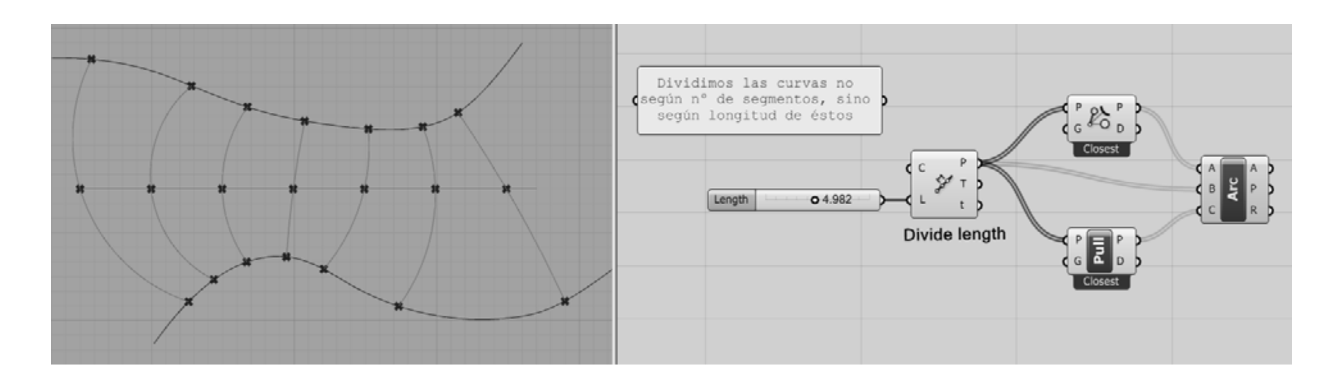




Con la orden "Pull Point" (en Vector/Point) proyectamos los puntos de una curva sobre otra (mínima distancia).
En la imagen anterior, los de una recta intermedia sobre las dos curvas entre las que se encuentra.

A continuación, con los puntos obtenidos podemos trazar, por ejemplo, arcos de circunferencia (”Arc 3Pt”, en Curve/Primitive).

Y podemos igualmente dividir las curvas no según el nº de segmentos, sino definiendo la longitud de éstos.



Hiperboloide hiperbólico

En este punto vamos a definir una superficie reglada, que pudiera parecer muy compleja, pero no lo es tanto, sobre todo si se crea a partir de sus generatrices, y nos va a permitir repasar algunas de las transformaciones básicas vistas hasta ahora. Se trata del hiperboloide hiperbólico.

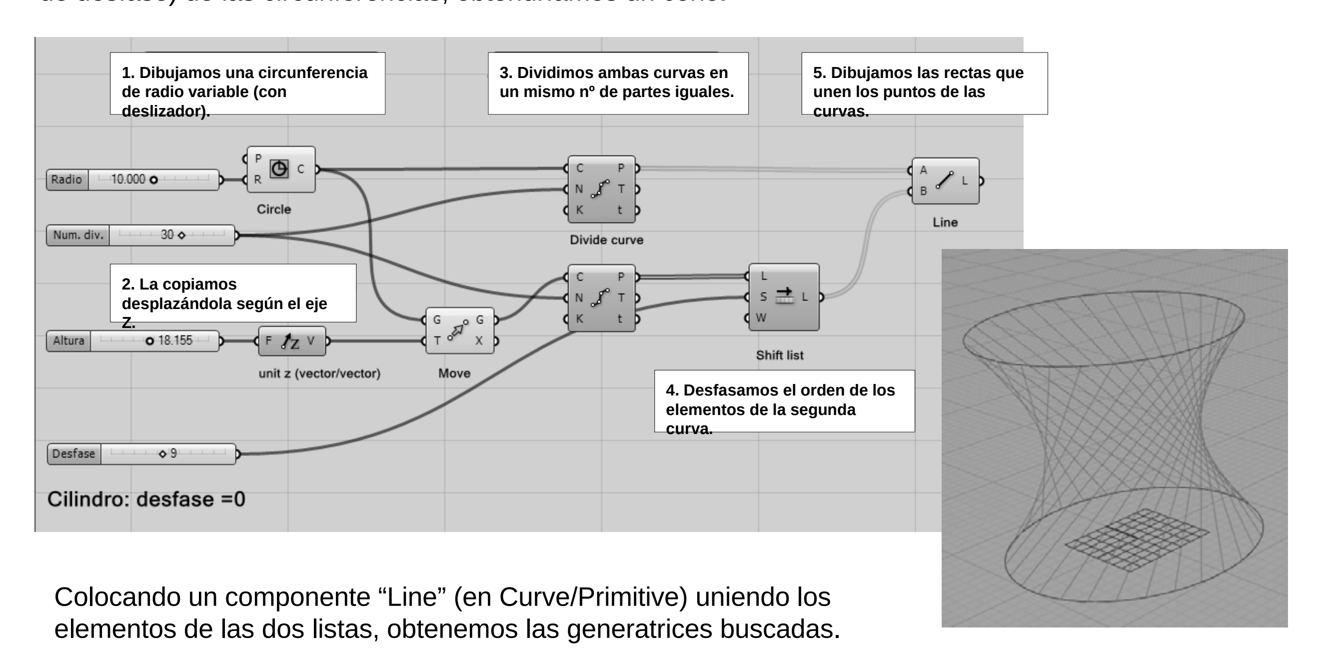
1. Comenzaremos dibujando la circunferencia de una de las bases (por su sencillez, vamos a dibujar un hiperboloide de eje vertical).

2. La desplazaremos (con el componente “Move”, que ya hemos visto) según el eje Z, colocando un deslizador para poder modificarla, una distancia igual a la altura del hiperboloide.

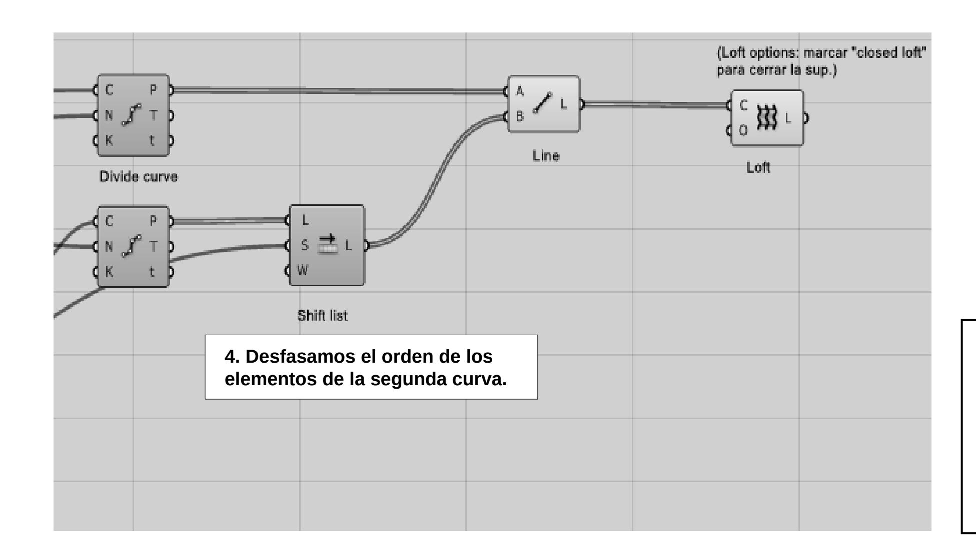
3. A continuación dividiremos ambas circunferencia, la original y su copia desplazada según Z, en un mismo número de partes (con el componente “Divide curve”, que acabamos de utilizar).

4. Los puntos de división de ambas curvas constituyen dos listas de datos. Pues bien, obtendremos las generatrices del hiperboloide sin más que desfasar el orden de los elementos de una de las dos listas respecto a la otra, antes de unir mediante rectas los puntos correspondientes.
Esto es precisamente lo que hace el componente “**Shift List**” (en Sets/List).

El desfase (parámetro S en “Shift list”, que controlaremos con un deslizador), determinará el ancho de garganta del hiperboloide. Si fuese nulo, obtendríamos un cilindro; y en caso de coincidir con la mitad del número de divisiones (que tendría que ser par, para que la mitad sea un número entero de pasos de desfase) de las circunferencias, obtendríamos un cono.



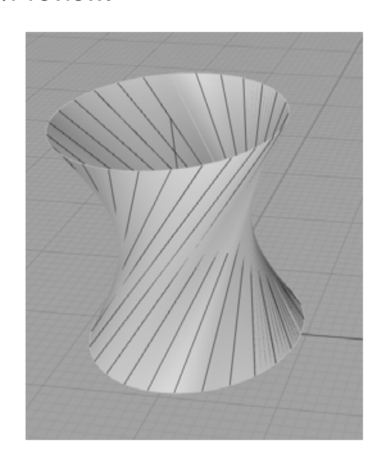
Colocando un componente “Line” (en Curve/Primitive) uniendo los elementos de las dos listas, obtenemos las generatrices buscadas.

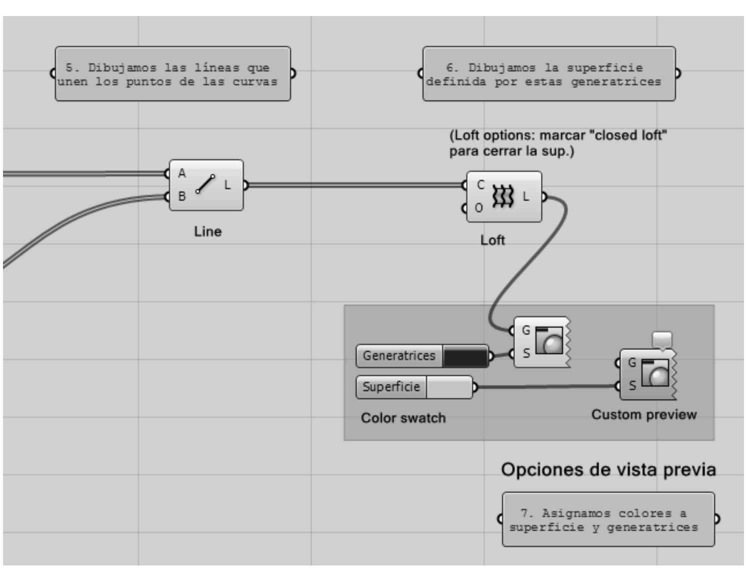


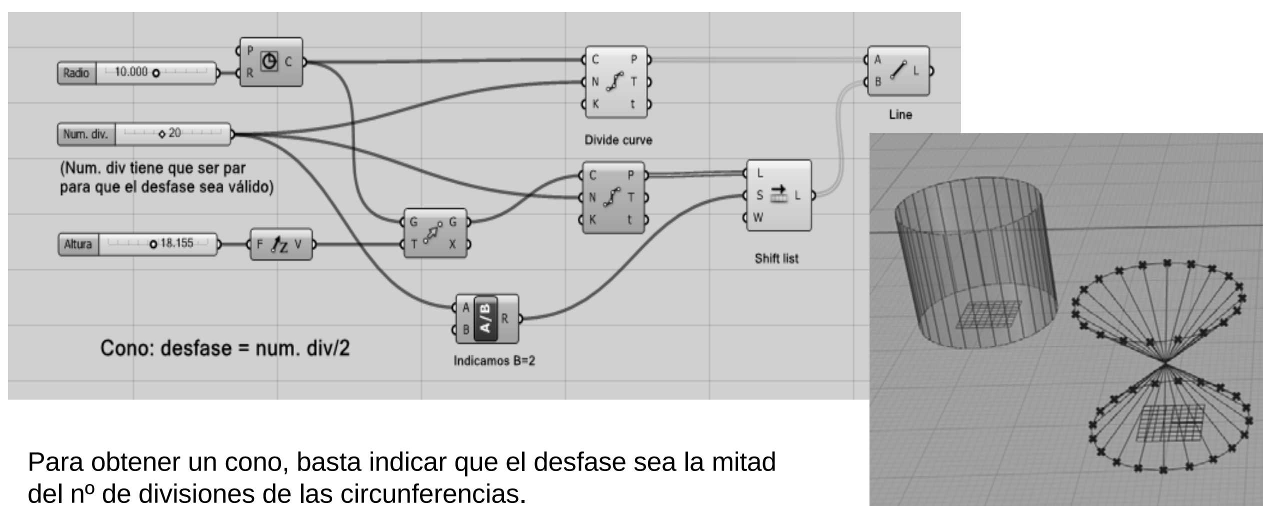
En las opciones de "Loft" ("Loft options" con btn. dcho. sobre la entrada "O" marcaremos la opción "Closed loft" para que la superficie quede cerrada.

Podemos finalmente dibujar la superficie definida por la red de generatrices. Para ello basta utilizar el componente "**Loft**" (en Surface/Transform).

Y podemos asignar colores, tanto a la superficie como a las generatrices, con el componente "Custom preview", al que asociaremos un "Color Swatch", ambos en Display/Preview.







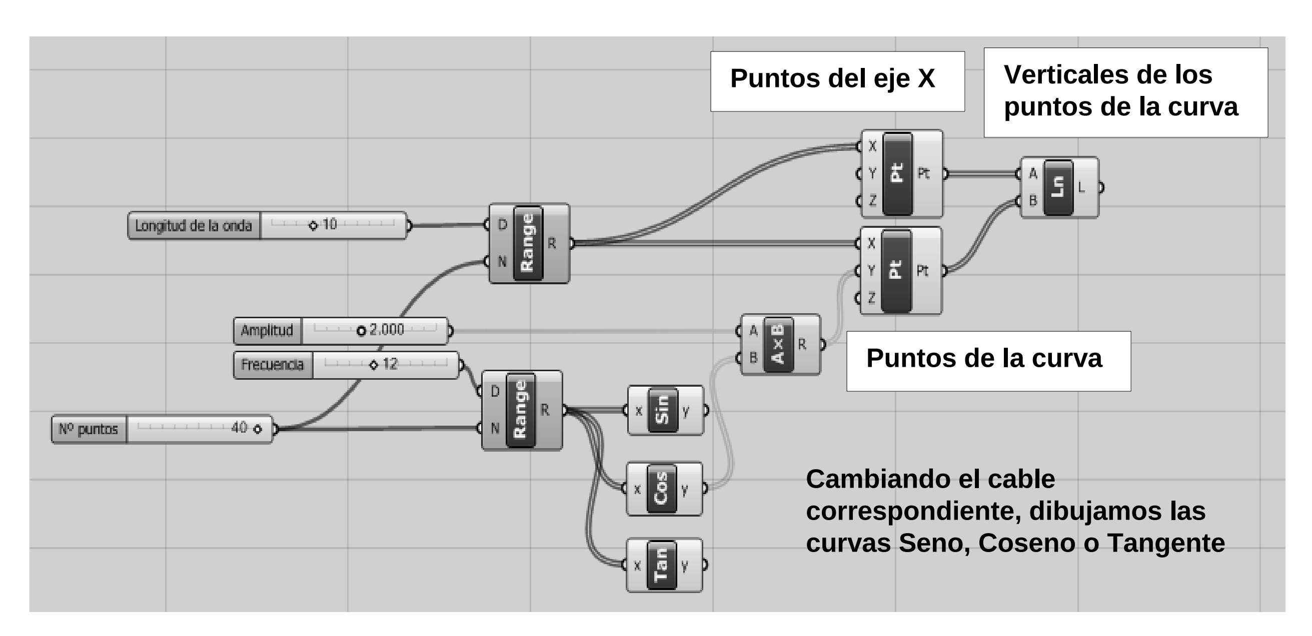
Para obtener un cono, basta indicar que el desfase sea la mitad del nº de divisiones de las circunferencias.

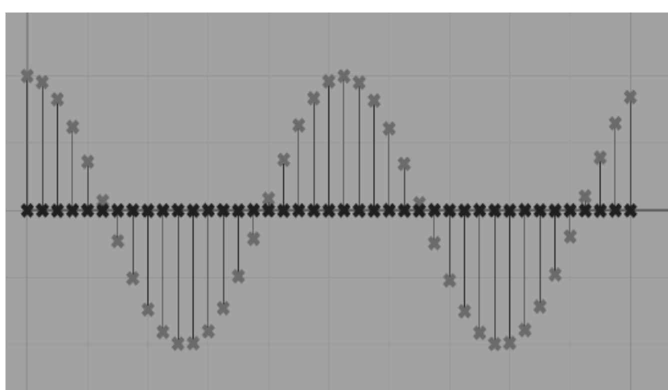
Curvas trigonométricas

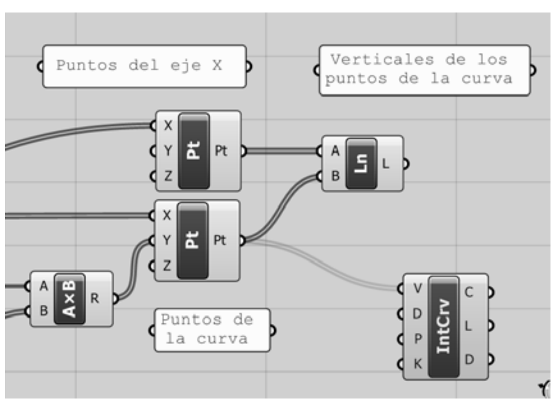
Vamos a terminar el capítulo de trazado de curvas manejando “Rangos” de valores, en este caso con curvas que pueden definirse matemáticamente, como son las curvas trigonométricas.

Puesto que vamos a empezar dibujando curvas planas, definiremos dos rangos de valores, uno para abscisas y otro para ordenadas.

Los puntos que determinen cada curva estarán comprendidos entre unos valores máximos y mínimos en X e Y, que vamos a identificar con lo que en la definición hemos denominado “frecuencia” y “longitud de las ondas”

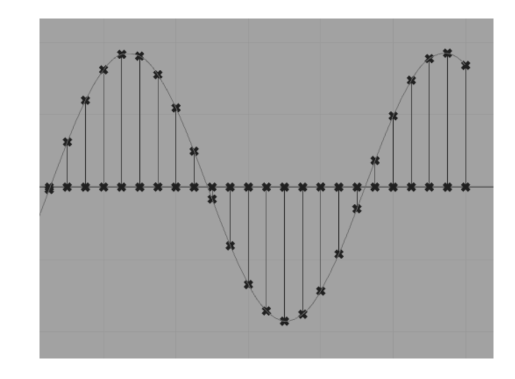




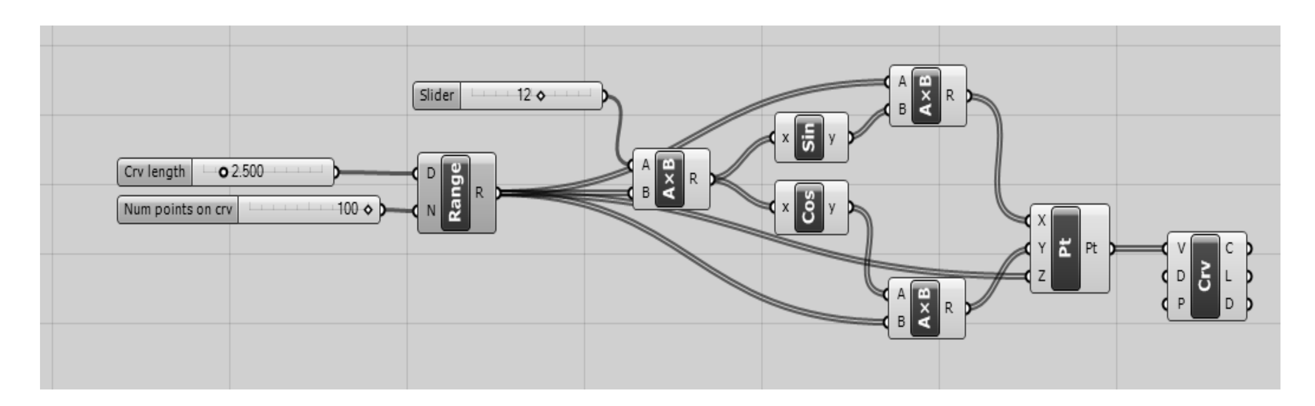


La definición que hemos descrito nos permite dibujar los puntos del eje X, los correspondientes a sus ordenadas (según utilicemos una u otra función), y las verticales que conectan unos con otros.

Finalmente dibujaremos la curva interpolando los puntos de la curva que hemos obtenido (componente “Interpolate” en Curve/Spline).



Hélice trigonométrica



A continuación vamos a definir una curva en tres dimensiones, utilizando fórmulas matemáticas, que nos va a permitir presentar una nueva forma de entrada de datos en Rhino: el **editor de expresiones**.

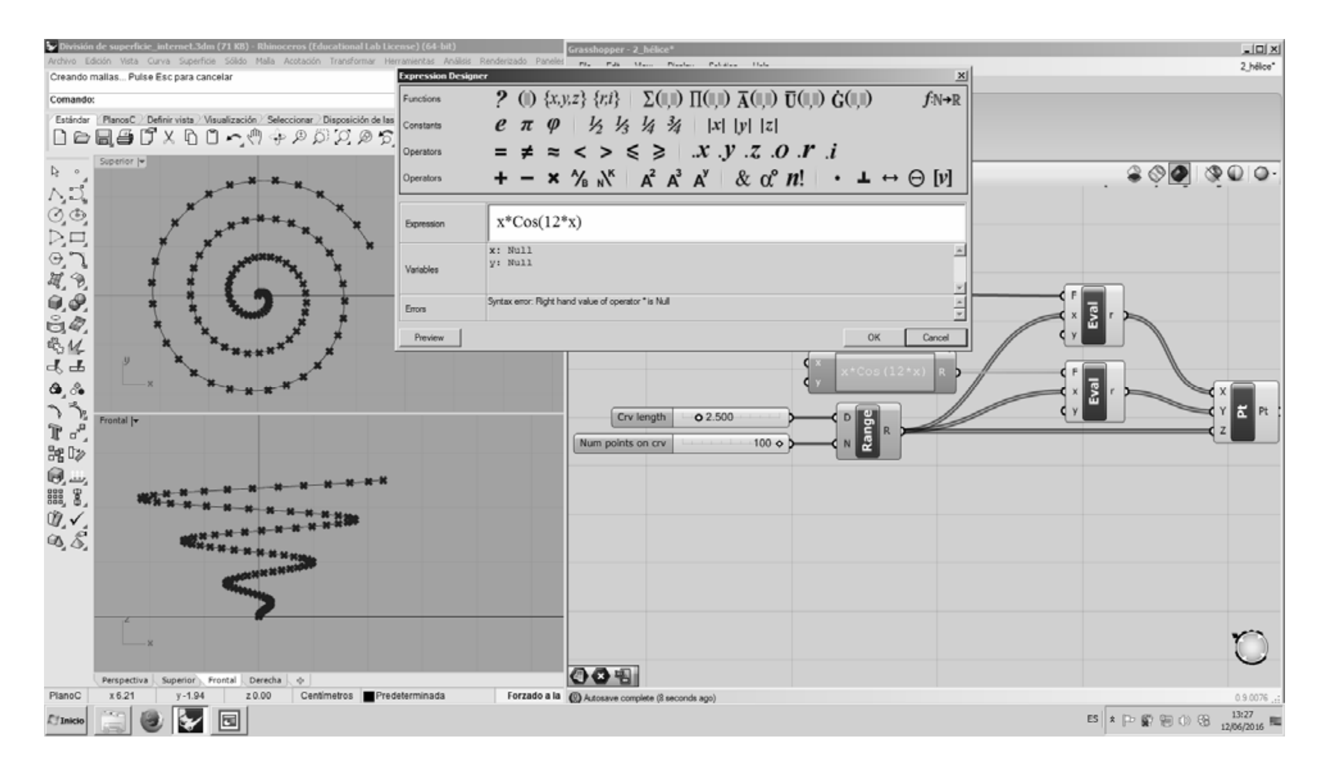
Se trata de una hélice cónica, cuyos puntos tienen coordenadas X, Y, Z definidas por las siguientes funciones:

X = AxSen (RxA); y = AxCos (RxA); Z = A

En nuestro ejemplo hemos utilizado un deslizador para que el valor de R, que representa la amplitud de la curva, sea variable (12 en la definición que incluimos).

La definición puede hacerse utilizando los componentes de Rhino (como en el ejemplo superior), o directamente, introduciendo el valor de las expresiones a aplicar.

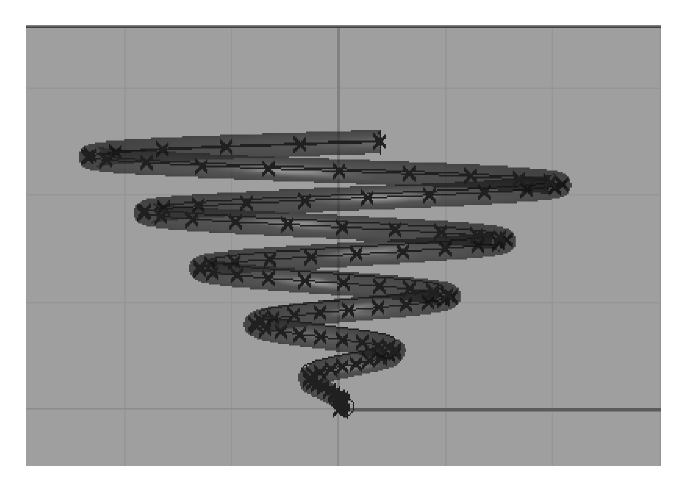
Con el botón derecho del ratón sobre el nombre de un componente seleccionamos "Expression" para abrir el editor de expresiones, que tiene el aspecto que vemos en la imagen anterior.

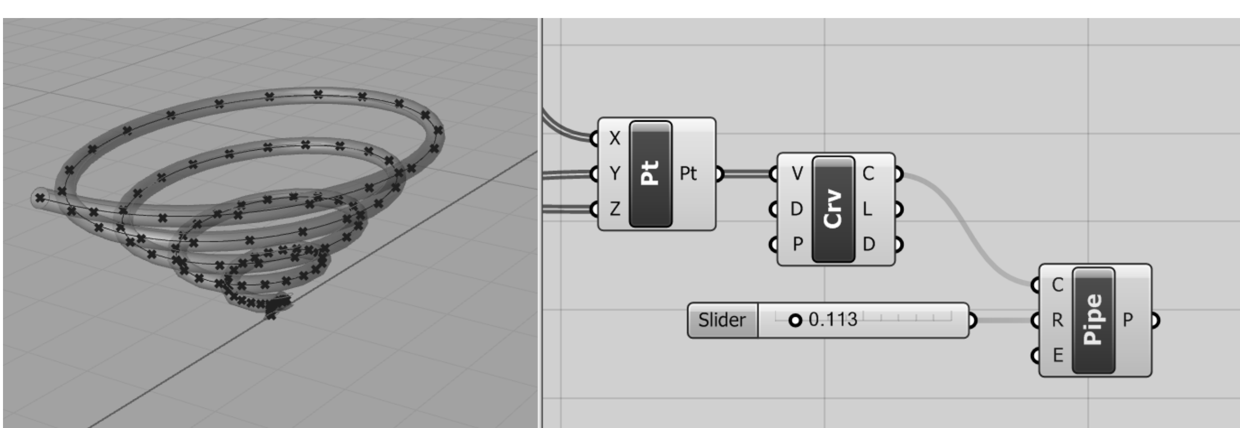


Tan solo necesitamos conectar nuestras expresiones a dos componentes de evaluación (“**Evaluate**”, en Maths/Script), que serán las entradas X e Y de nuestros puntos.

Dibujamos finalmente una tubería a lo largo de la curva, un elemento que resulta muy útil para ayudar a destacar líneas en la visualización, y que utilizaremos con frecuencia en nuestras definiciones.

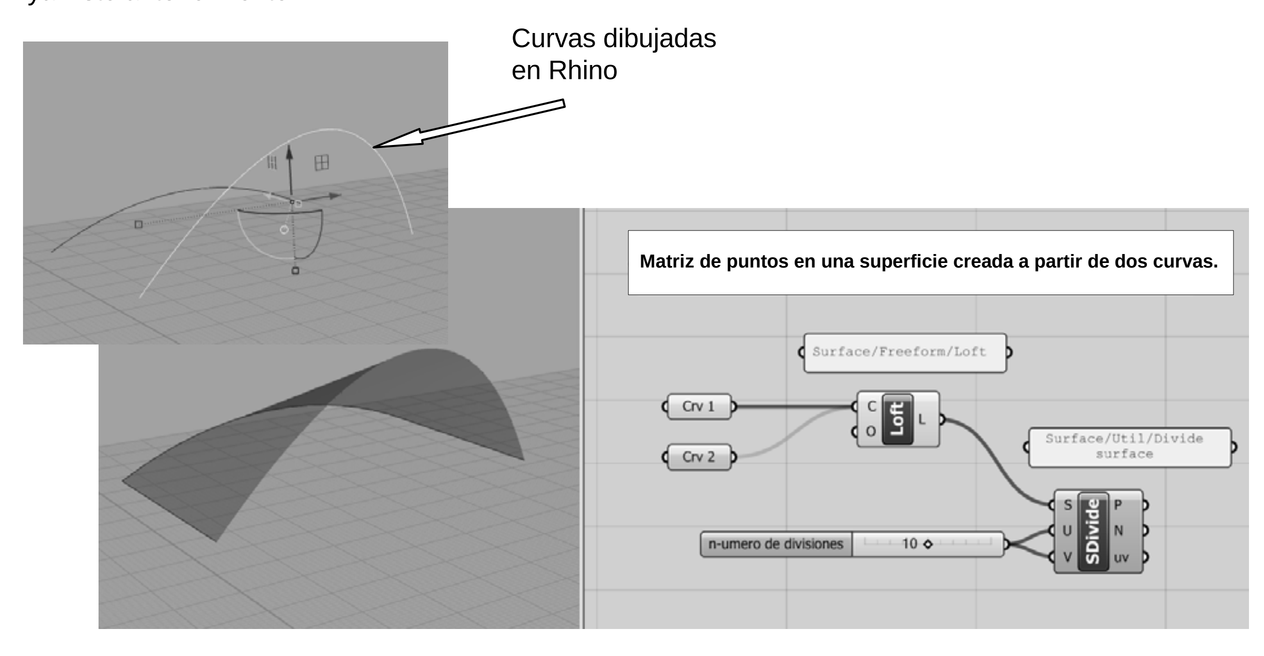
El componente correspondiente se denomina “**Pipe**”, en Surface/Freeform, y requiere tan sólo la curva (parámetro C), el radio de la tubería (R), y el modo en que queremos cerrar los extremos (E, por defecto sin cerrar, pero podemos escoger tapas planas o esféricas).





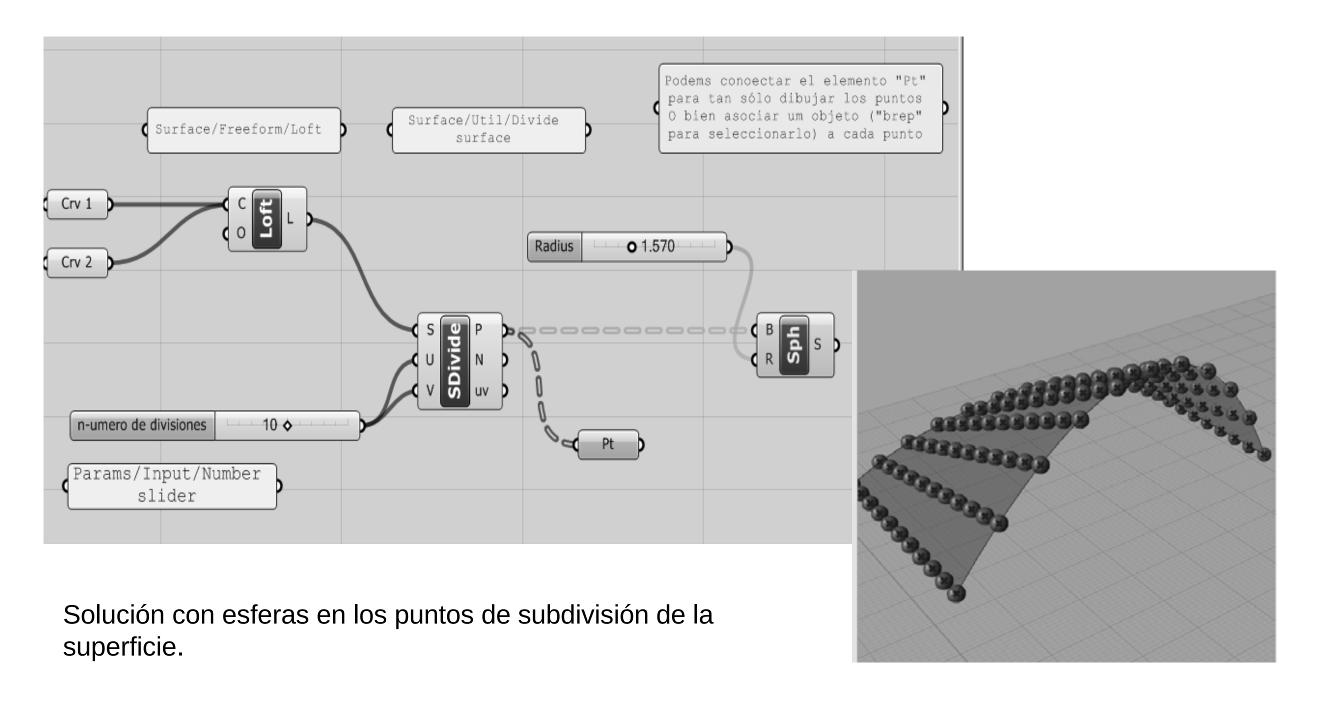
Superficies. Matriz de puntos creada en una superficie generada a partir de dos curvas

A partir de dos curvas 3D creadas en Rhino, definimos una superficie utilizando el componente “Loft”, ya visto anteriormente.

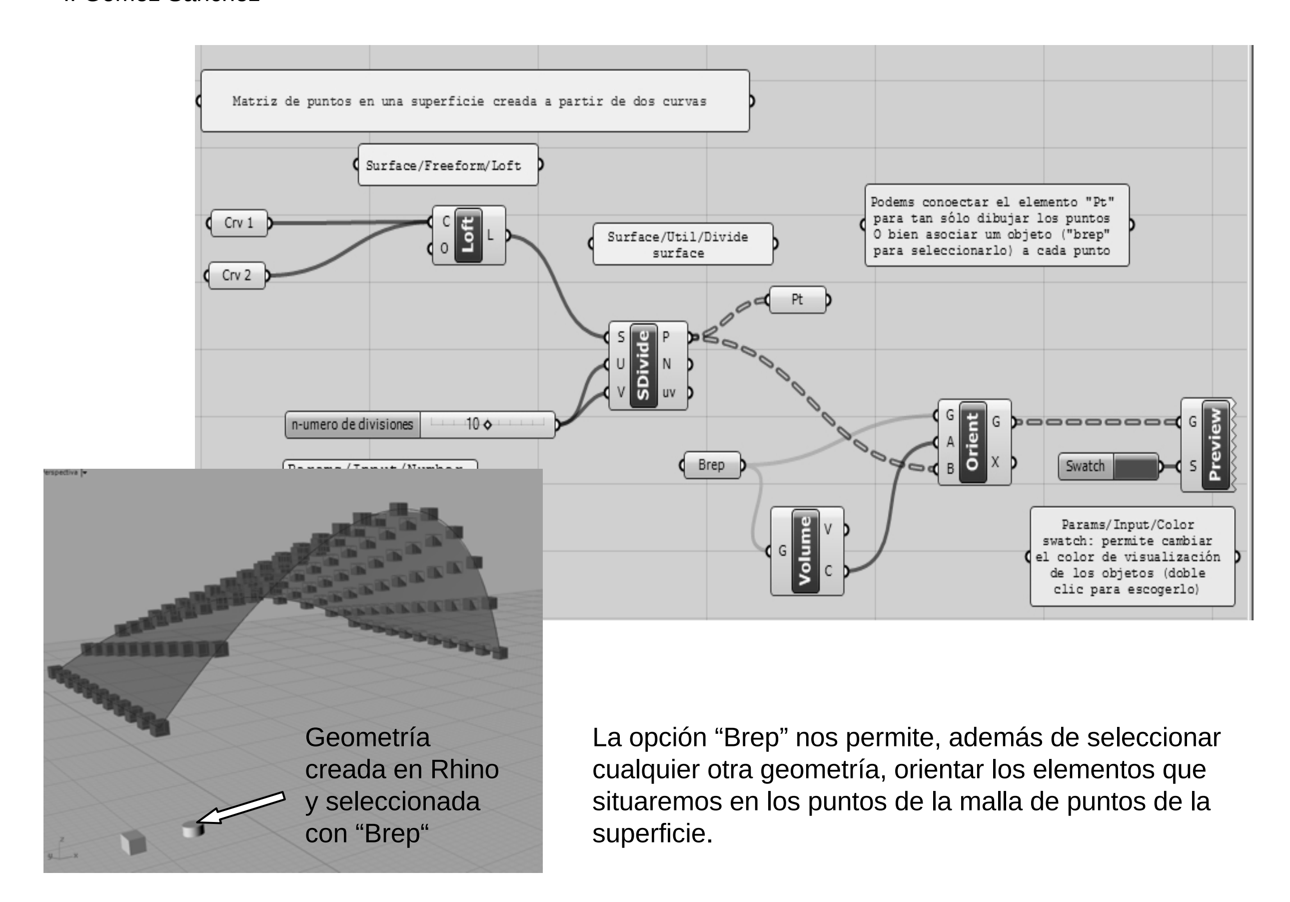


A continuación dividimos la superficie (“Divide Surface”, *SDivide*, en Surface/Util), según dos componentes “u” y “v”; en este caso, en un mismo número de divisiones en cada una de ellas (pues hemos conectado el mismo deslizador a ambas).

Podemos colocar una esfera en cada uno de los puntos de división. O bien otra entidad geométrica cualquiera, que seleccionaremos incluyendo un componente intermedio denominado **Brep** (“Boundary REPresentation”, en Params/Geometry).



Solución con esferas en los puntos de subdivisión de la superficie.



Geometría creada en Rhino y seleccionada con “Brep“

La opción “Brep” nos permite, además de seleccionar cualquier otra geometría, orientar los elementos que situaremos en los puntos de la malla de puntos de la superficie.

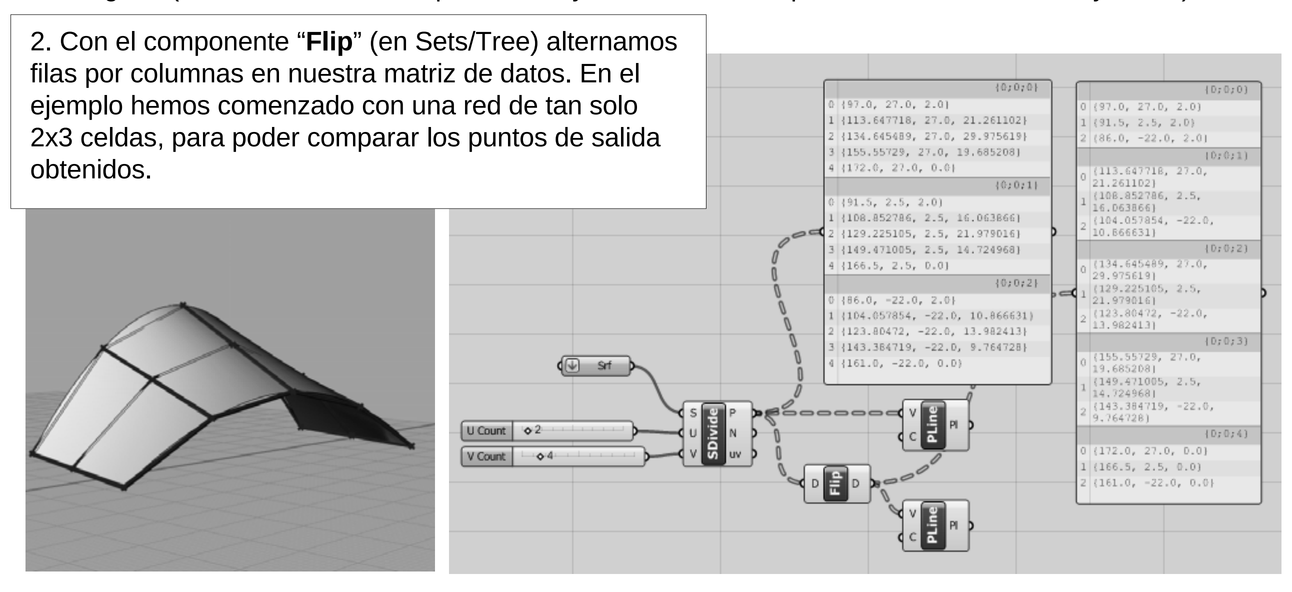
División de superficies. Mallado ortogonal

A continuación vamos a explicar una sencilla forma de dividir superficies en celdas perpendiculares entre sí, que nos va a resultar muy útil para panelar superficies.

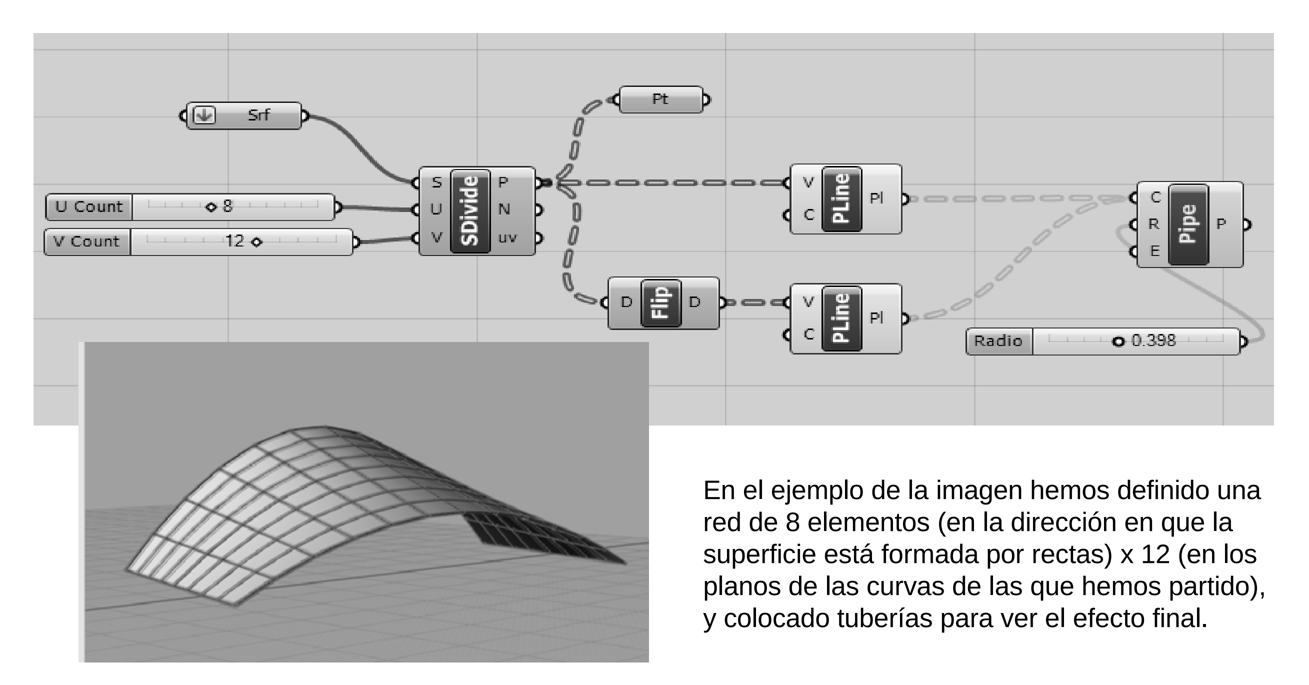
Veamos la definición de la rutina creada:

1. Subdividimos una superficie que previamente habremos creado en Rhino y seleccionado en GH (con el componente “SDivide”, que acabamos de ver, y en este caso definimos una red no cuadrada sino rectangular (con deslizadores independientes y valores distintos para las divisiones en “u” y en “v”).

2. Con el componente “**Flip**” (en Sets/Tree) alternamos filas por columnas en nuestra matriz de datos. En el ejemplo hemos comenzado con una red de tan solo 2x3 celdas, para poder comparar los puntos de salida obtenidos.



Con “Flip” obtenemos las series de datos correspondientes a dos direcciones ortogonales de nuestra red, por lo que para dibujar las líneas de la malla que definen basta enlazar dos componentes “**PolyLine**” (en Curve/Spline), y a su vez a éstos simples líneas o tuberías, para visualizar los elementos lineales de nuestra malla ortogonal.



En el ejemplo de la imagen hemos definido una red de 8 elementos (en la dirección en que la superficie está formada por rectas) x 12 (en los planos de las curvas de las que hemos partido), y colocado tuberías para ver el efecto final.

Triangulación de superficies. Manejo de datos

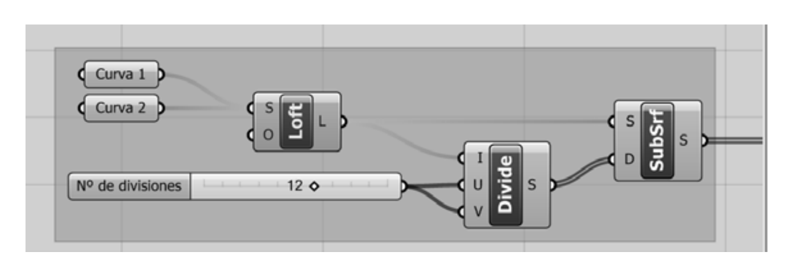
En próximas páginas continuaremos viendo mallados automáticos de superficies según dos direcciones, pero antes vamos a hacer un paréntesis que nos va a permitir entender la estructura de datos de nuestras redes.

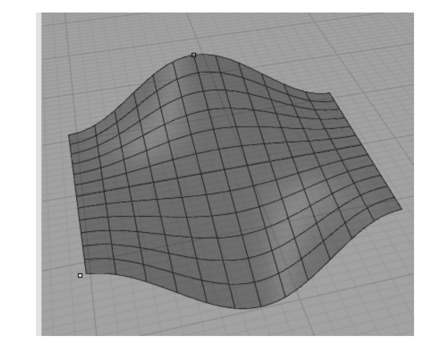
Nuestro objetivo será dibujar una red, no rectangular sino triangular, en una superficie cualquiera (que o bien habremos creado previamente en Rhino, o bien crearemos desde GH a partir de curvas de Rhino, como ya hemos visto en los ejemplos anteriores).

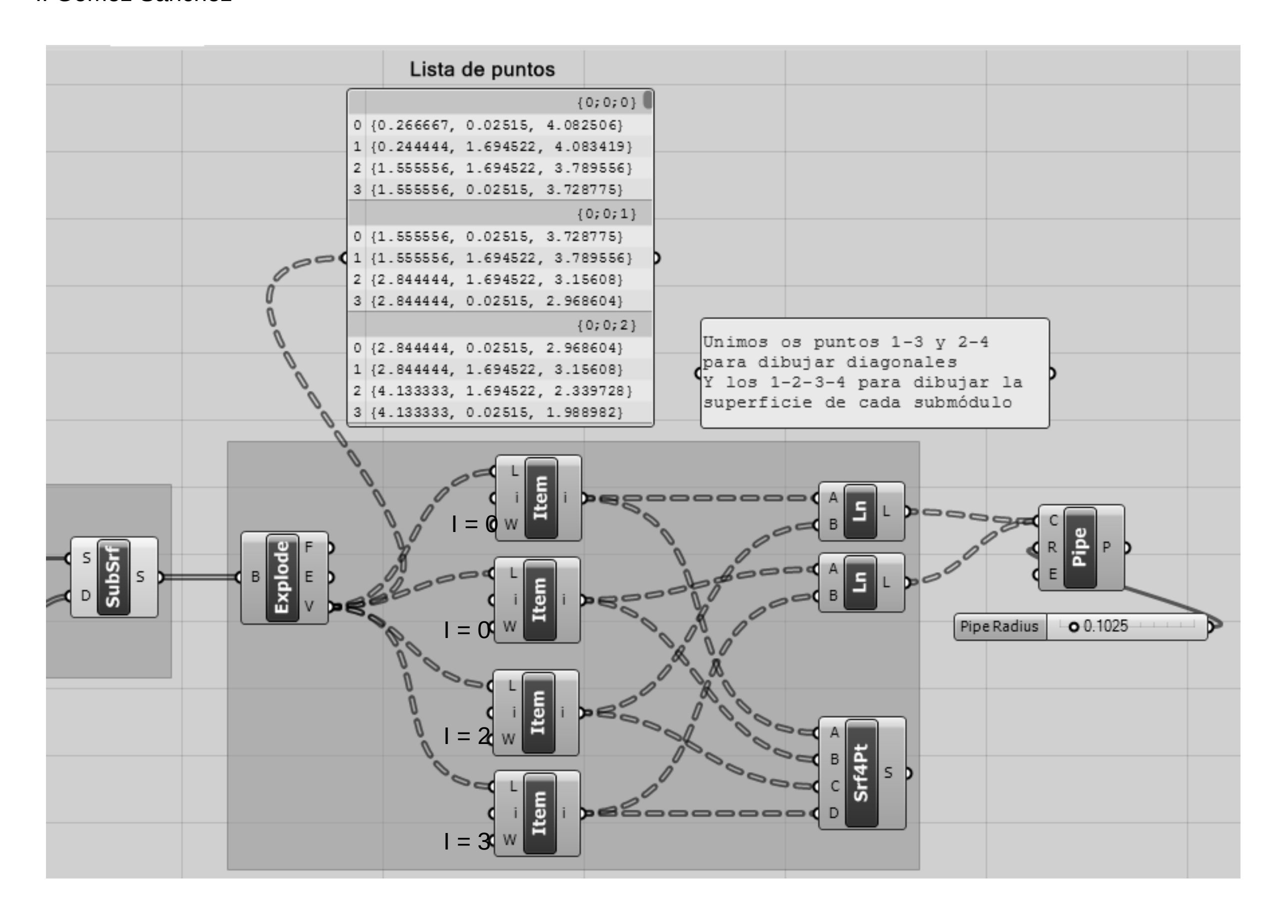
1. Partiremos de una superficie alabeada cualquiera (como la que en el ejemplo hemos generado a partir de dos curvas mediante la ya conocida orden “Loft”).

En primer lugar generaremos un dominio de puntos con los que resulten de dividirla según dos direcciones “u” y “v” (que pueden tener el mismo número de elementos, como en nuestro ejemplo, o diferente). Utilizaremos el componente “Divide Domain” (*Divide*), en Maths/Domain.

Con estos puntos determinaremos las series de 4 elementos que forman las subsuperficies, utilizando el componente “**Isotrim**” (*SubSrf*), en Surface/Util, y extraeremos las coordenadas ordenadas en bloques de 4 elementos mediante la orden “Deconstruct Brep” (***Explode***), en Surface/Analysis.

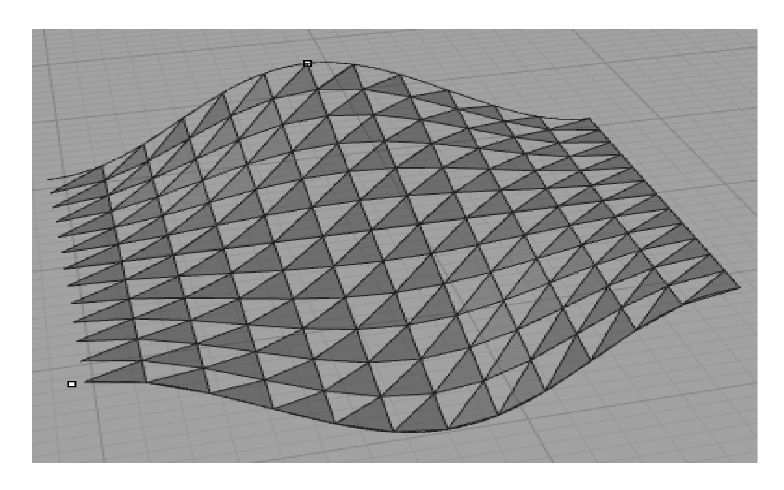


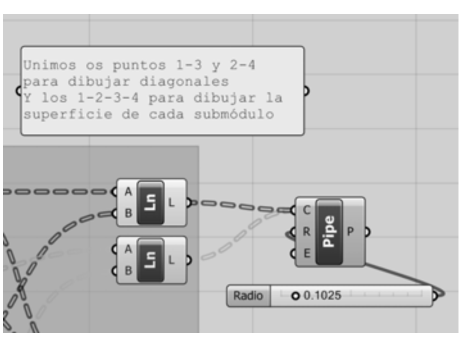


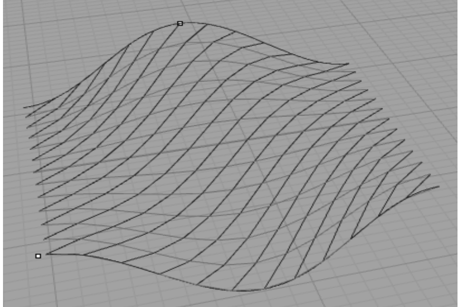


Como vemos en el panel de puntos, cada grupo de cuatro coordenadas va del 0 al 3. Para construir áreas triangulares, podemos emplear el componente “4Point Surface” (*Srf4P*t), en Surface/Freeform, uniendo los elementos 0, 0, 2 y 3 de cada grupo.

Es lo que hemos hecho al asignar estos valores a la entrada “i” (índice) de cada elemento, y a continuación asociarlos a los 4 puntos de *Srf4Pt*.





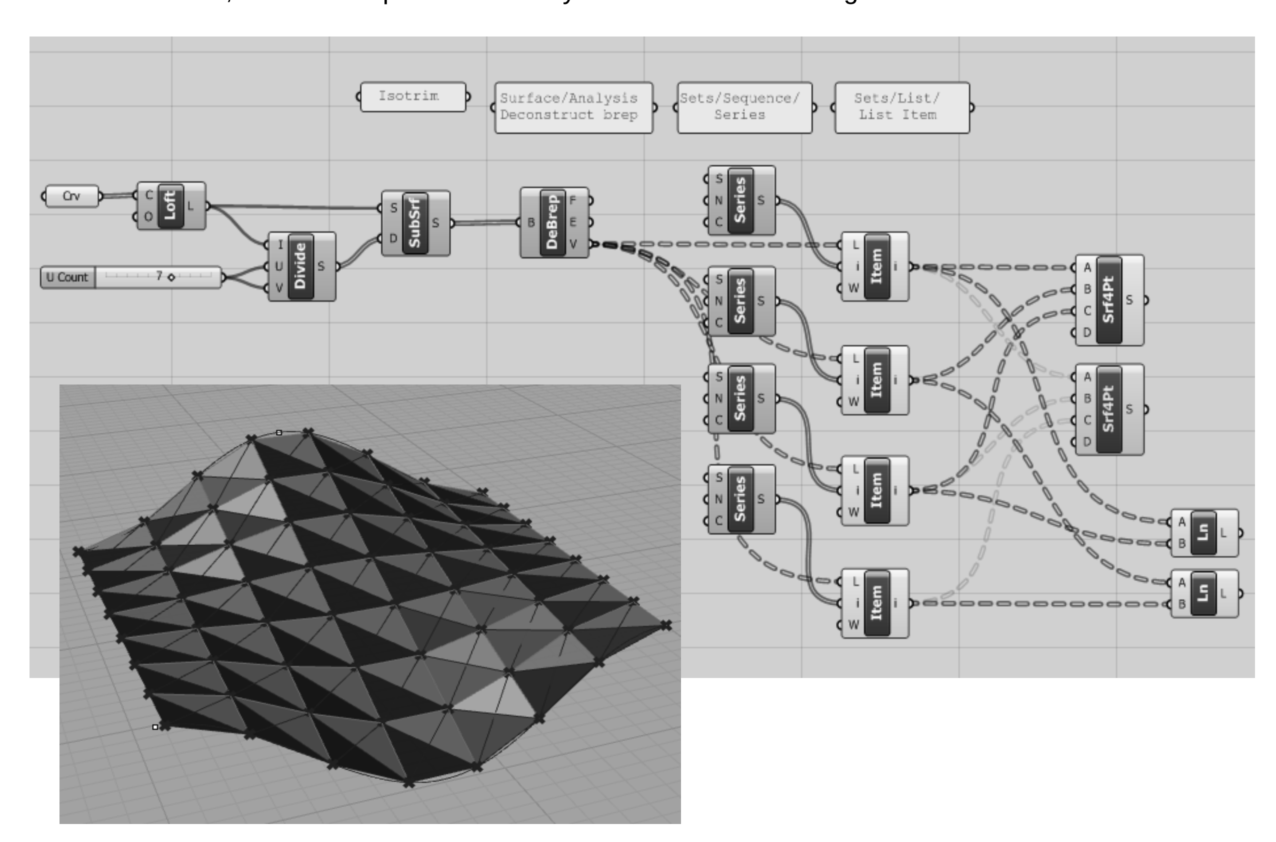


Podemos dibujar los elementos lineales de nuestra malla, y colocar tuberías a lo largo de ellos:

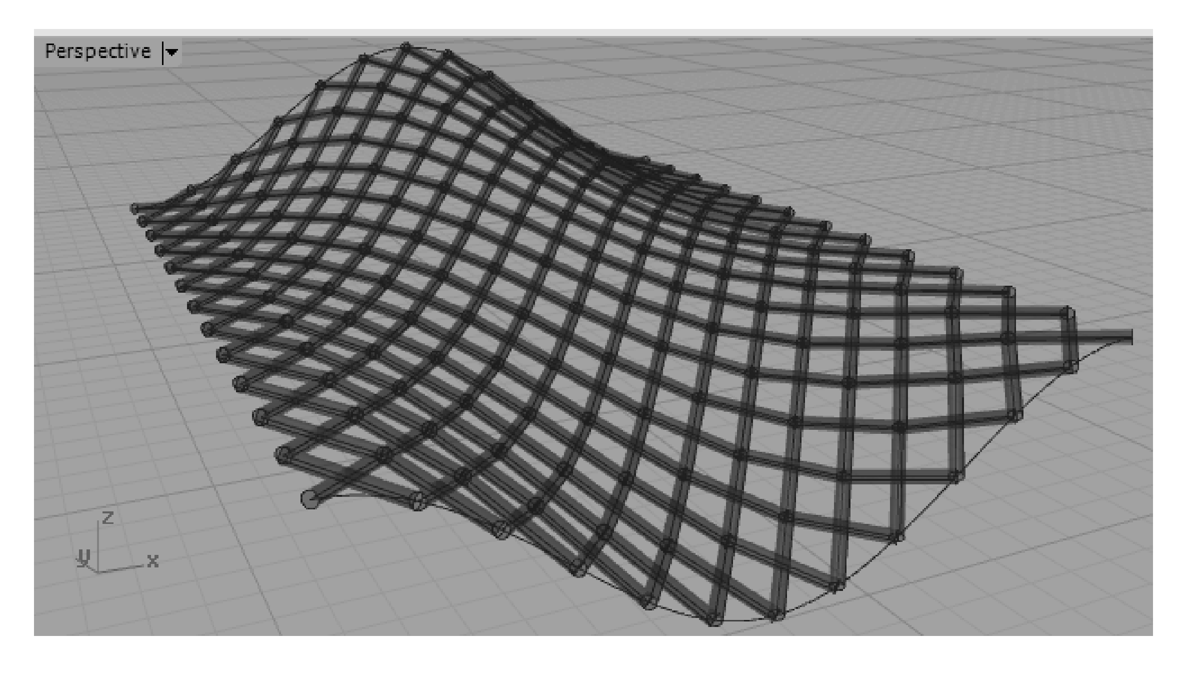
- Con el 1º y el 3º de cada grupo dibujaremos las diagonales;
- Con el 2º y el 4º las líneas paralelas a los planos de las curvas.

El componente “Item” permite extraer de la lista de coordenadas obtenidas tras “deconstruir” las subsuperficies con “Explode”, las que tienen el índice que indiquemos.

También podemos dibujar las dos familias de superficies triangulares, como hemos hecho en esta nueva definición, uniendo los puntos 1-2-3-1 y 1-3-4-1 de cada subregión:



O bien determinar las líneas de la estructura, dependiendo de si nos interesa trabajar con los planos para, por ejemplo, panelarlos, o si lo que pretendemos es resolver los elementos resistentes.   .

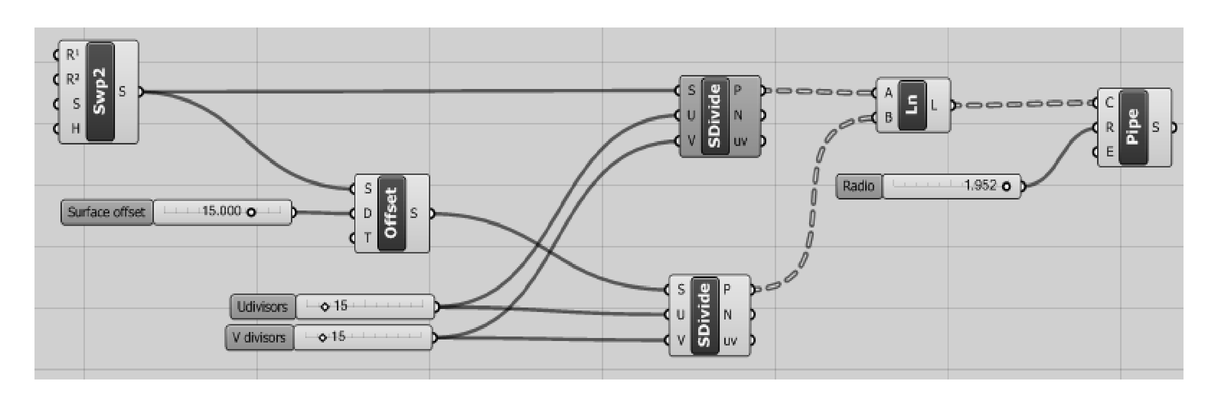


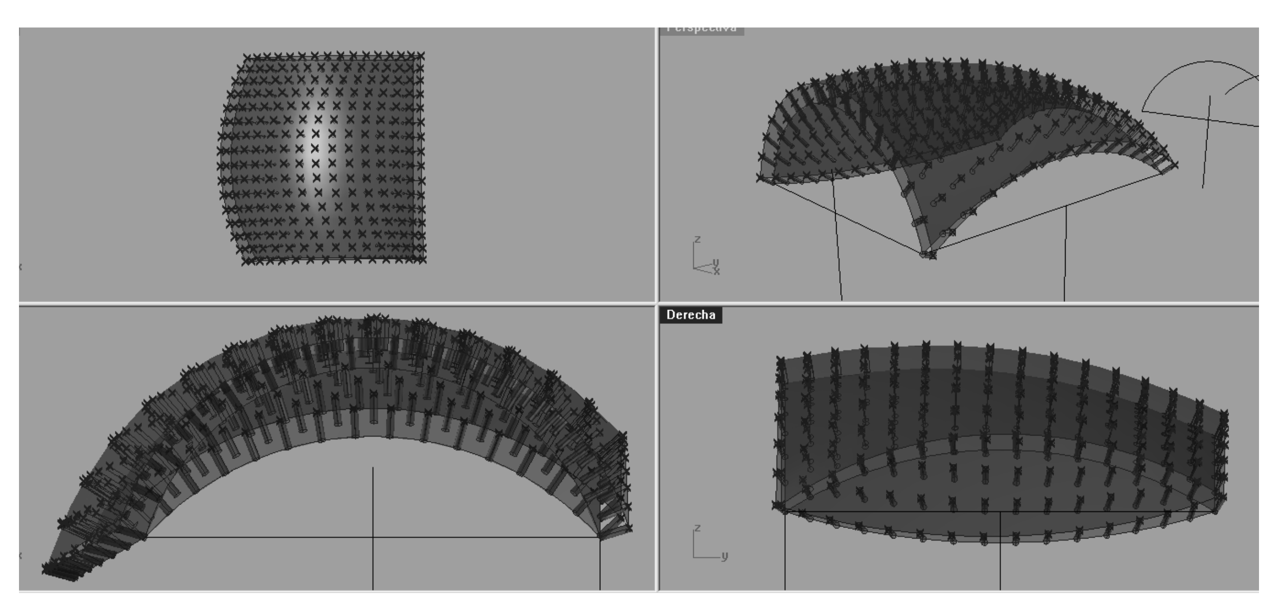
Conexión de superficies

También podemos diseñar superficies de doble capa. Basta para ello con hacer un “**Offset**” (en Surface/ Util) de una superficie para copiarla a determinada distancia.

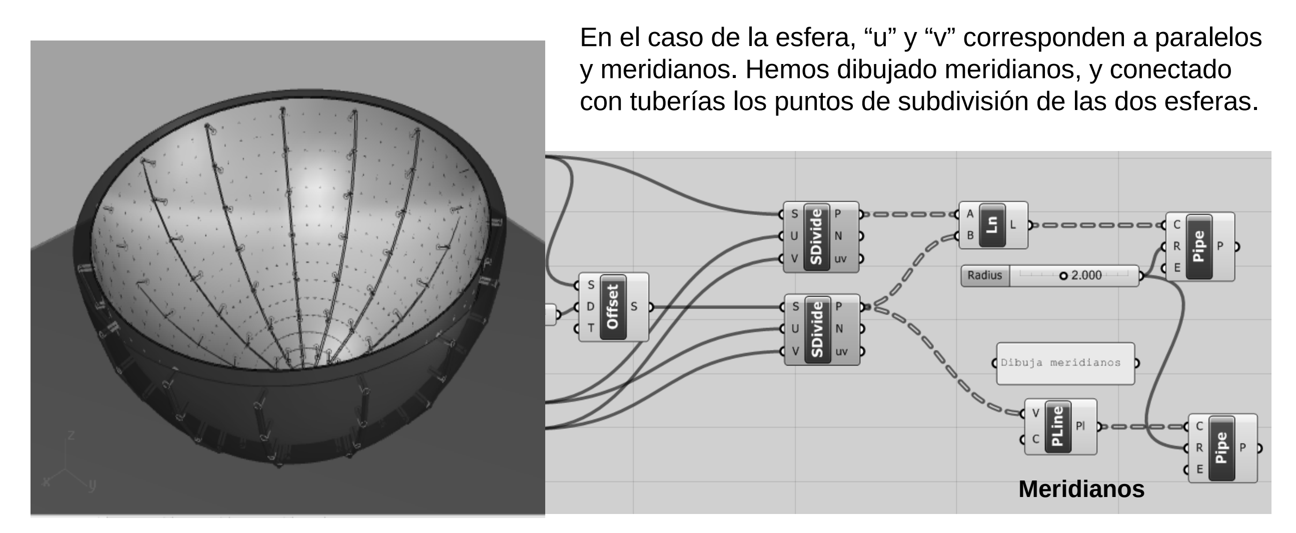
Podremos asimismo colocar conectores entre ellas, sin más que añadir tuberías (“Pipe”) a lo largo de lineas que unan los puntos de subdivisión de ambas.

Lo vemos en la siguiente definición:





En el caso de la esfera, “u” y “v” corresponden a paralelos y meridianos. Hemos dibujado meridianos, y conectado con tuberías los puntos de subdivisión de las dos esferas.



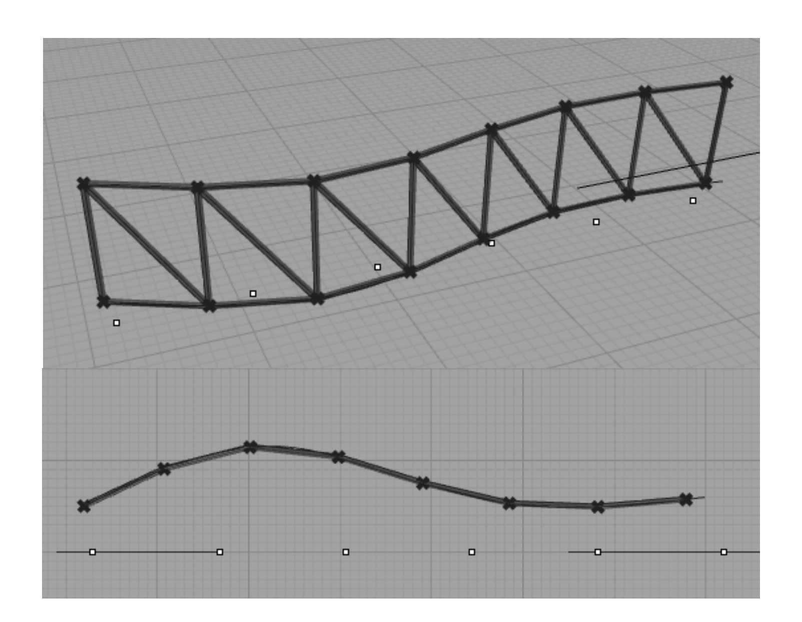
Cercha básica

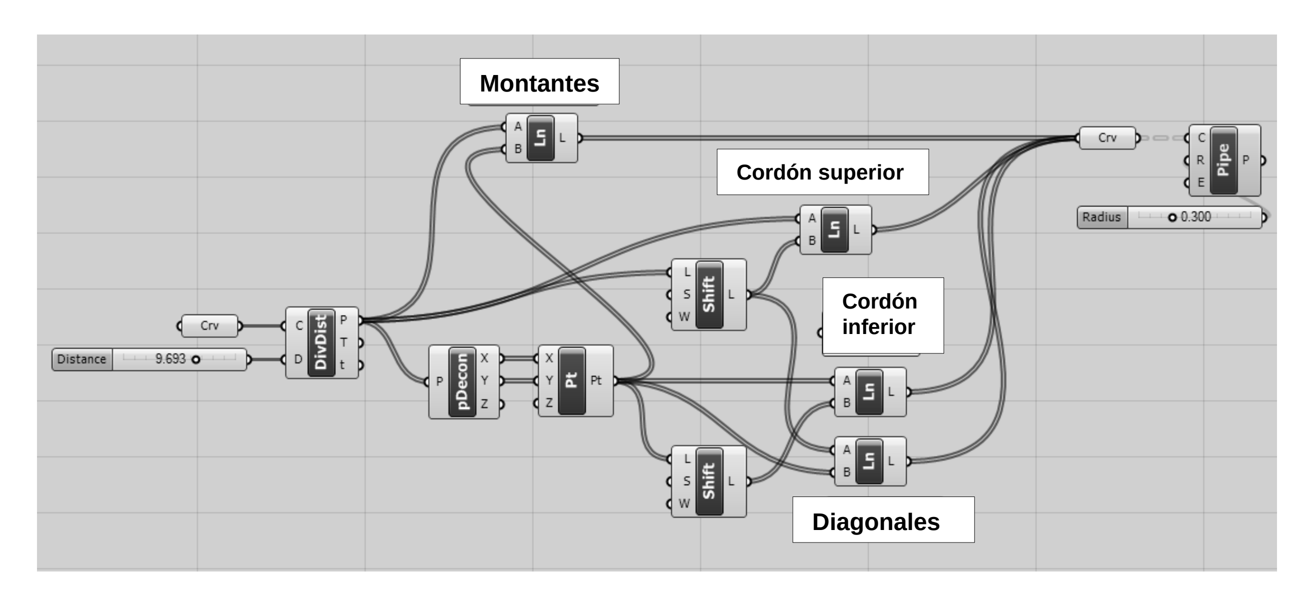
Finalmente vamos a aplicar el control del orden de los datos de nuestras listas para conectarlos de manera que determinen las líneas de una cercha básica a lo largo de una línea.

1. Empezamos dividiendo la curva en tramos de una longitud determinada, con “Divide Distance” (*DivDist*), en Curve/Division.

2. Extraemos las coordenada X, Y, Z de los puntos de división, con “Deconstruct+t” (*pDecon*, en Vector/Point.

3, Construimos de nuevo los puntos con sólo coordenadas X e Y, para determinar la proyección sobre la curva inferior.





Según utilicemos unos u otros estaremos trabajando con los puntos del cordón superior o del inferior respectivamente.

Como ya sabemos, el componente “Shift” nos permite desfasar los elementos de las listas. Desfasamos una unidad los puntos de ambos cordones:

- Si unimos los de cada lista obtenemos los cordones superior e inferior;
- Si unimos el de una lista con el desfasado de la otra obtenemos las diagonales.

Morphing

Una de las aplicaciones más utilizadas en GH es quizá el panelado de superficies. Y muy especialmente la aplicación de patrones a las mismas. La potencia de cálculo de Rhinoceros permite ajustar geometrías bi o tridimensionales a superficies alabeadas.

1. Dividimos la superficie en subregiones

2. Construimos una red de cajas apoyadas en las subregiones

3. Ajustamos un patrón geométrico 3d a cada una de las caja

Escogemos la geometría definida en Rhino para el panelado de las cajas

Transform/Morph/Box morph

Seleccionamos una superficie construida en Rhino

3. Aplicamos a cada una de las cajas la geometría seleccionada.

Altura de cajas 2.000

U Count 15

V Count 13

2. Construimos cajas con base en las subregiones de la superficie

1.2. Subdividimos la superficie

1.1. Definimos un dominio de 2 variables para dividir la superficie

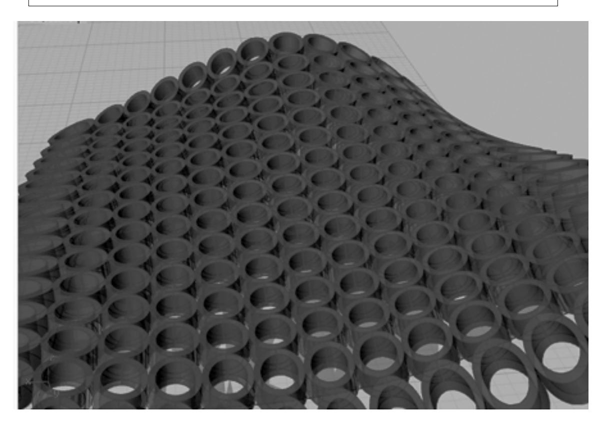
Podemos adaptar texturas tanto planas como tridimensionales a superficies, sin más que escoger un patrón y aplicarlo a subregiones de las mismas o a cajas tridimensionales definidas sobre estas subregiones respectivamente.

Todos los componentes y opciones correspondientes se encuentran en el menú Transform/Morph.

Denominamos “morphing” a la operación por la cual se aplica un patrón a la red de elementos de dos o tres dimensiones construida sobre una superficie, a la que nos estamos refiriendo.

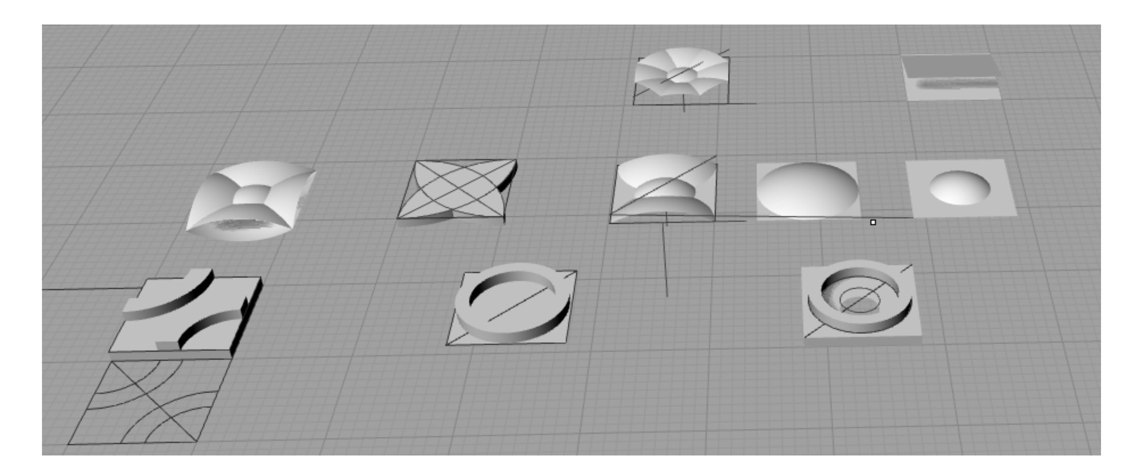


Morphing aplicando un módulo cilíndrico hueco.

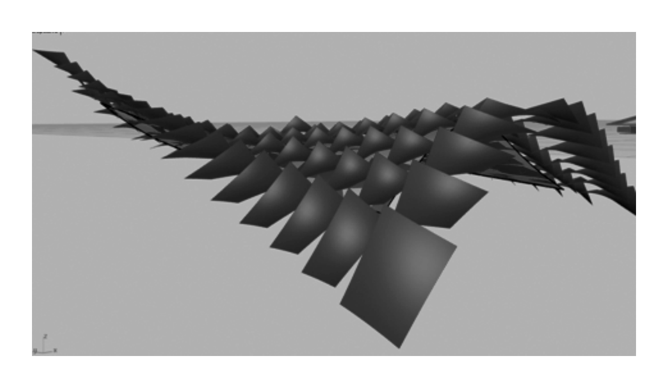


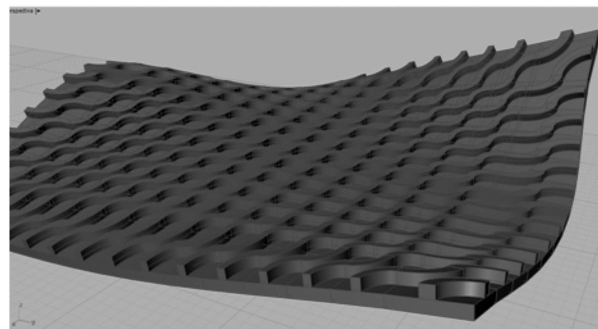
La definición necesaria para programar esta operación es muy sencilla. Consiste básicamente en:

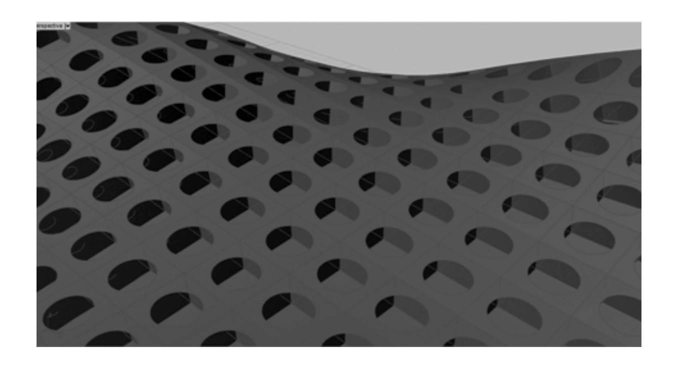
1. Dividir la superficie en subregiones (con el conocido “Isotrim” (*SubSrf)*, en Surface/Util;
2. Construir sobre cada una de ellas las cajas que harán las veces de contenedores (“Surface Box”, en Transform/Morph);
3. Ajustar el patrón seleccionado a estas cajas, que se adaptará a su geometría y proporciones.

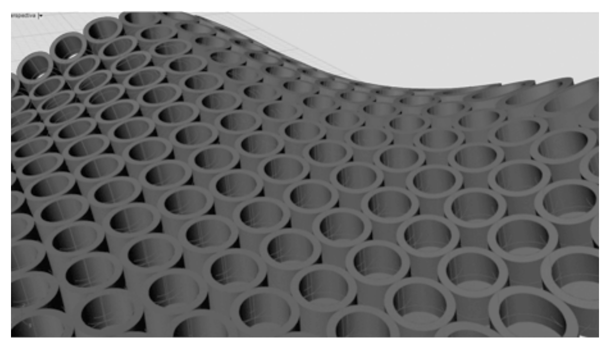


Geometrías definidas para el panelado con contenedores 3D (“Surface Box”, *SBox*, en Transform/Morph) de nuestro modelo.

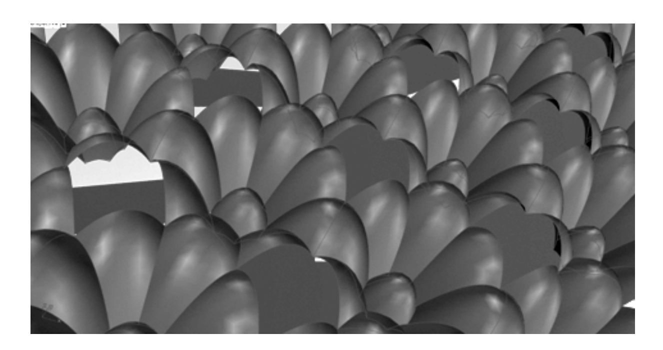


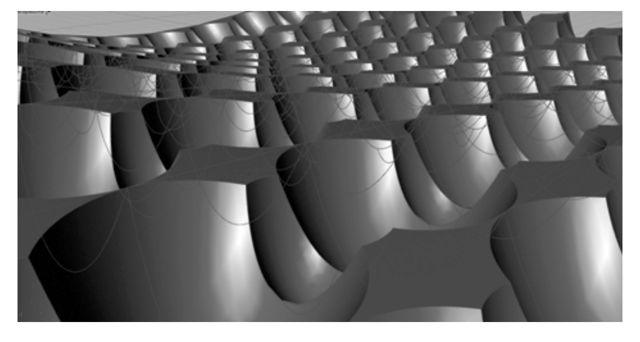






Las geometrías utilizadas para ajustar a nuestras cajas pueden ser polisuperficies, simples superficies o incluso elementos formados por líneas. Cualquier geometría 3D es válida.



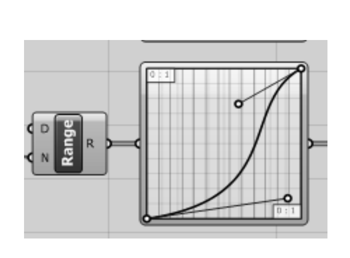


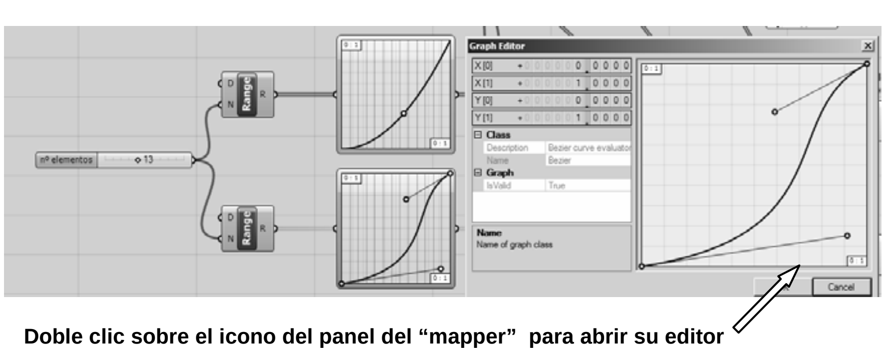
Panelado irregular

La distribución de filas y columnas de las redes definidas en una superficie no tiene por qué ser uniforme.

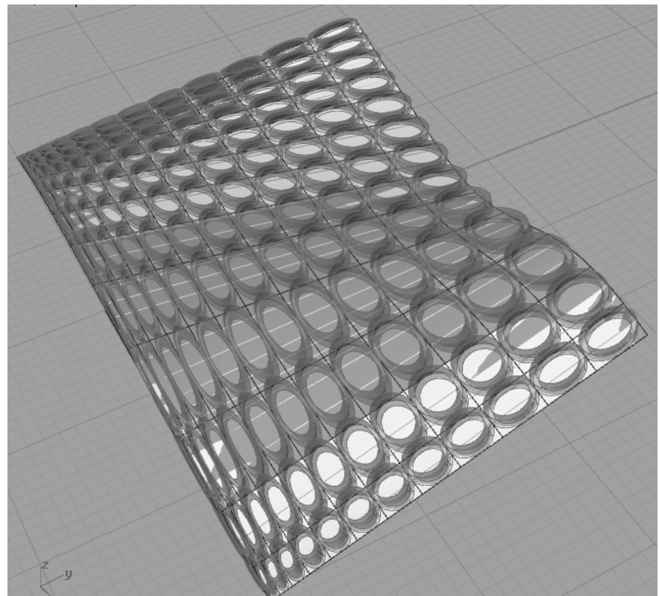
Podemos definir el tamaño de los elementos de cada una de las dos familias de coordenadas que generan las subregiones de manera diferente, y con variaciones a lo largo de las mismas.

Para ello resulta muy útil el “**Graph mapper**”, editor gráfico de distribución de la “topografía” de nuestras redes (en Params/Input). Permite definir las curvas de reparto de los dominios de datos.






Doble clic sobre el icono del panel del “mapper” para abrir su editor

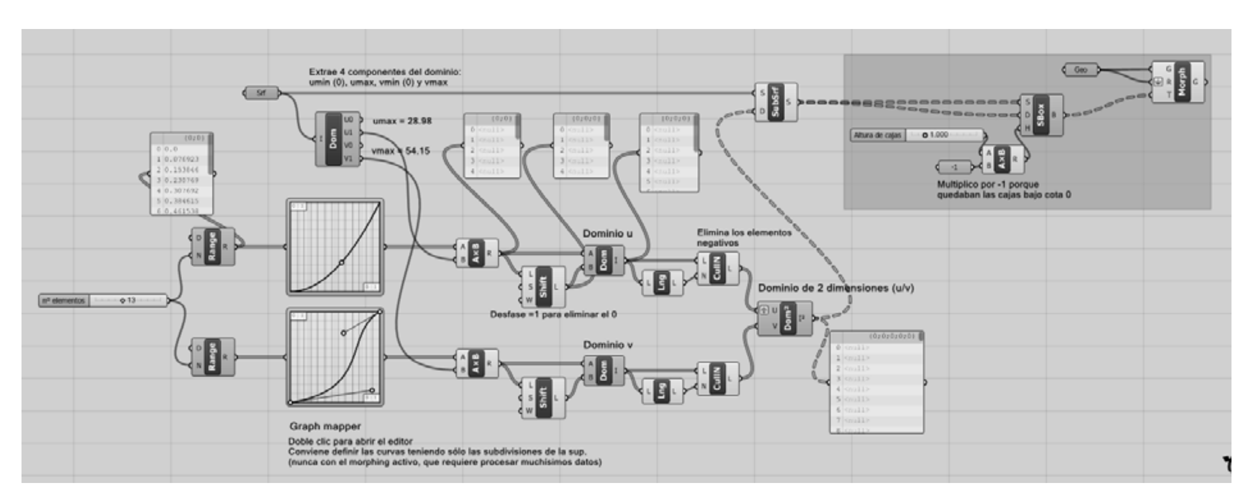


1. En nuestro ejemplo hemos definido dominios independientes para los parámetros “u” y “v” de una superficie; hemos determinado dos curvas de reparto del tamaño de los elementos de estas coordenadas, utilizando el editor gráfico de distribuciones;

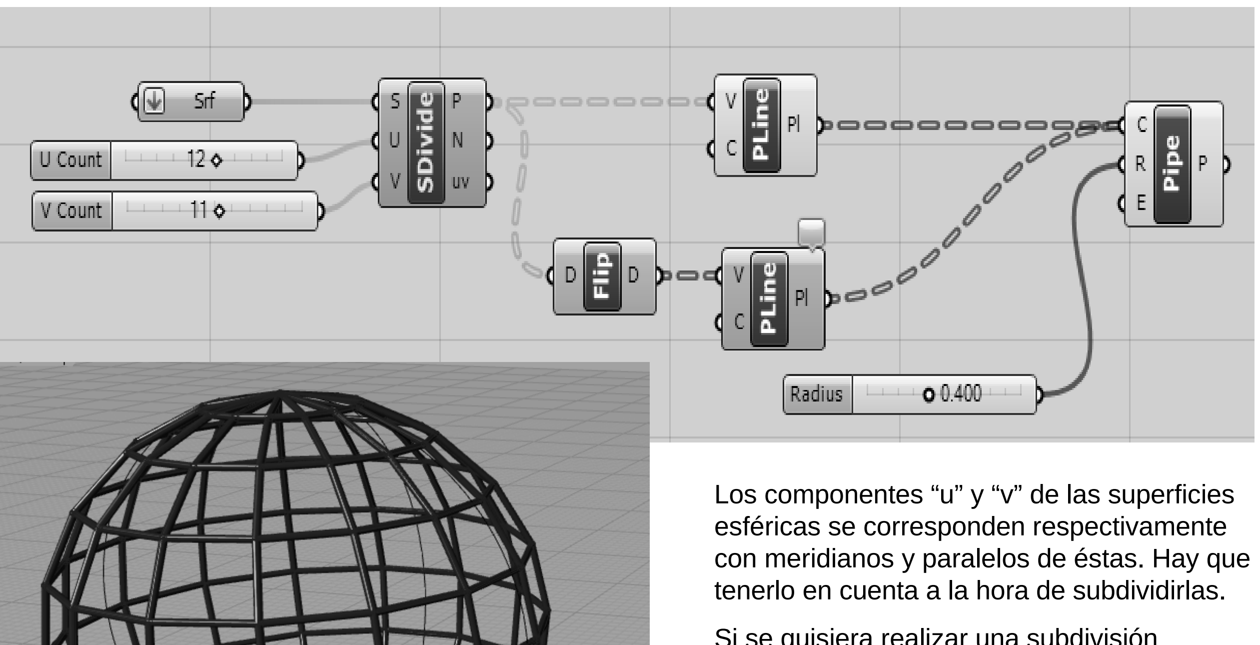
2. Tras hallar los valores mínimo y máximo y eliminar de las listas de datos los elementos negativos y el cero, hemos construido un dominio de dos dimensiones (u/v) combinando los dos unidimensionales (“Domain^2”, en Maths/domain);

3. Con este dominio hemos subdividido la superficie, con “Isotrim”;

4. Y finalmente hemos aplicado a las subregiones el “morphing” de la misma manera que en el caso general.

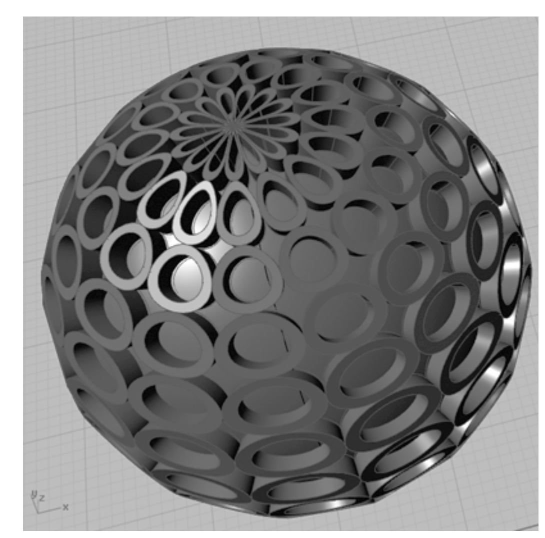


Panelado de esferas

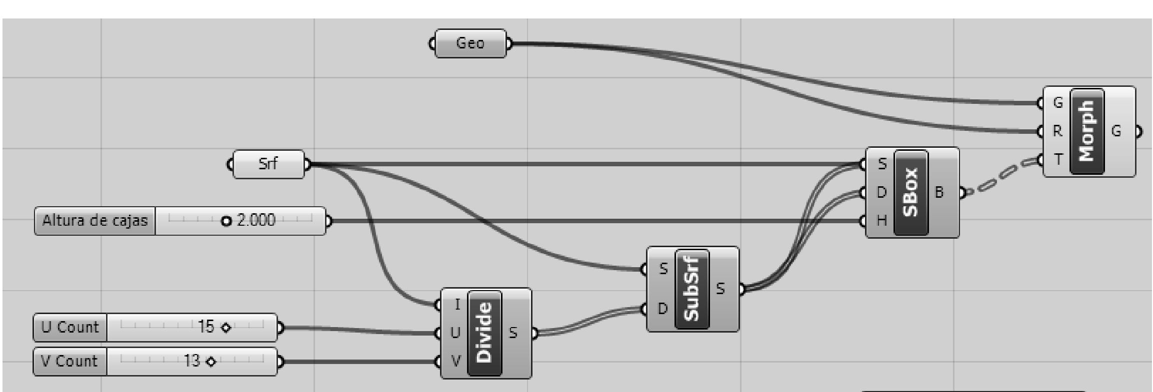


Los componentes “u” y “v” de las superficies esféricas se corresponden respectivamente con meridianos y paralelos de éstas. Hay que tenerlo en cuenta a la hora de subdividirlas.

Si se quisiera realizar una subdivisión homogénea, habría que trabajar con coordenadas polares con centro en el de la superficie.

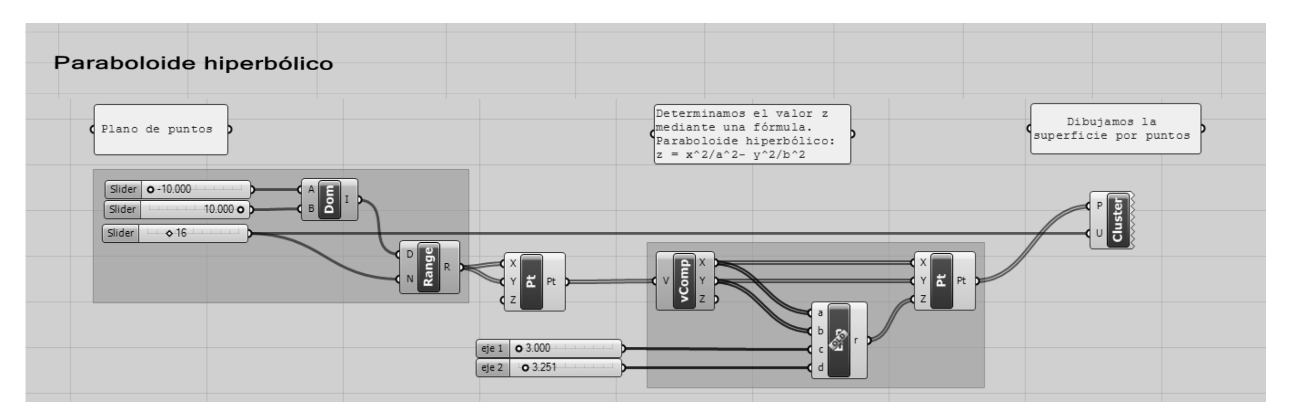


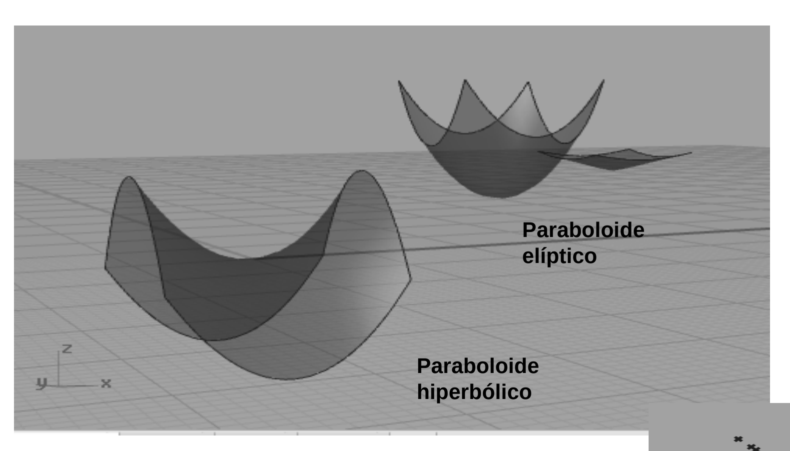
Y tenerlo en cuenta tanto si nuestro objetivo es definir una estructura de barrar sobre la superficie (véanse la imagen y la definición anteriores), como muy especialmente si pretendemos aplicar un patrón con el componente “Morph”, ya que éste se deformará adaptándose a esta red y quedando el patrón muy deformado en los polos.



Superficies mediante fórmulas

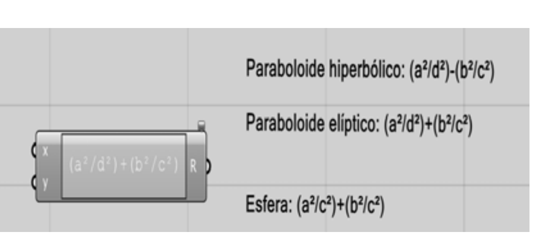
De la misma manera que hicimos para definir la geometría de una curva introduciendo su expresión matemática con el editor de expresiones, haremos esto mismo en el caso de las superficies definidas mediante fórmulas, con el componente “Expresion”, en *Maths/Script*.

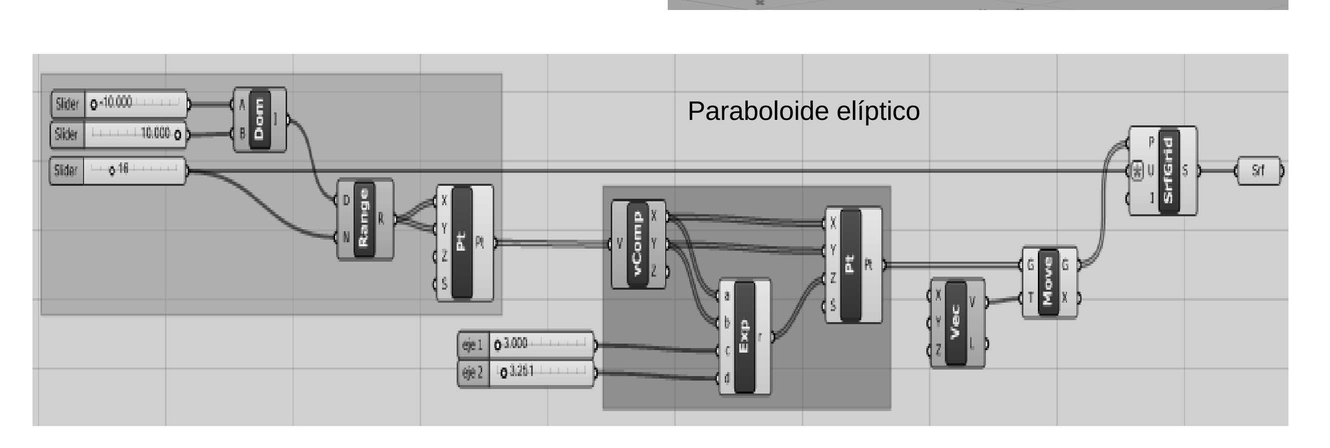




Por supuesto, y como siempre, podremos elegir entre utilizar el editor o construir la correspondiente expresión conectando componentes de GH. Esta segunda opción suele ser bastante más compleja, pero es igualmente válida.

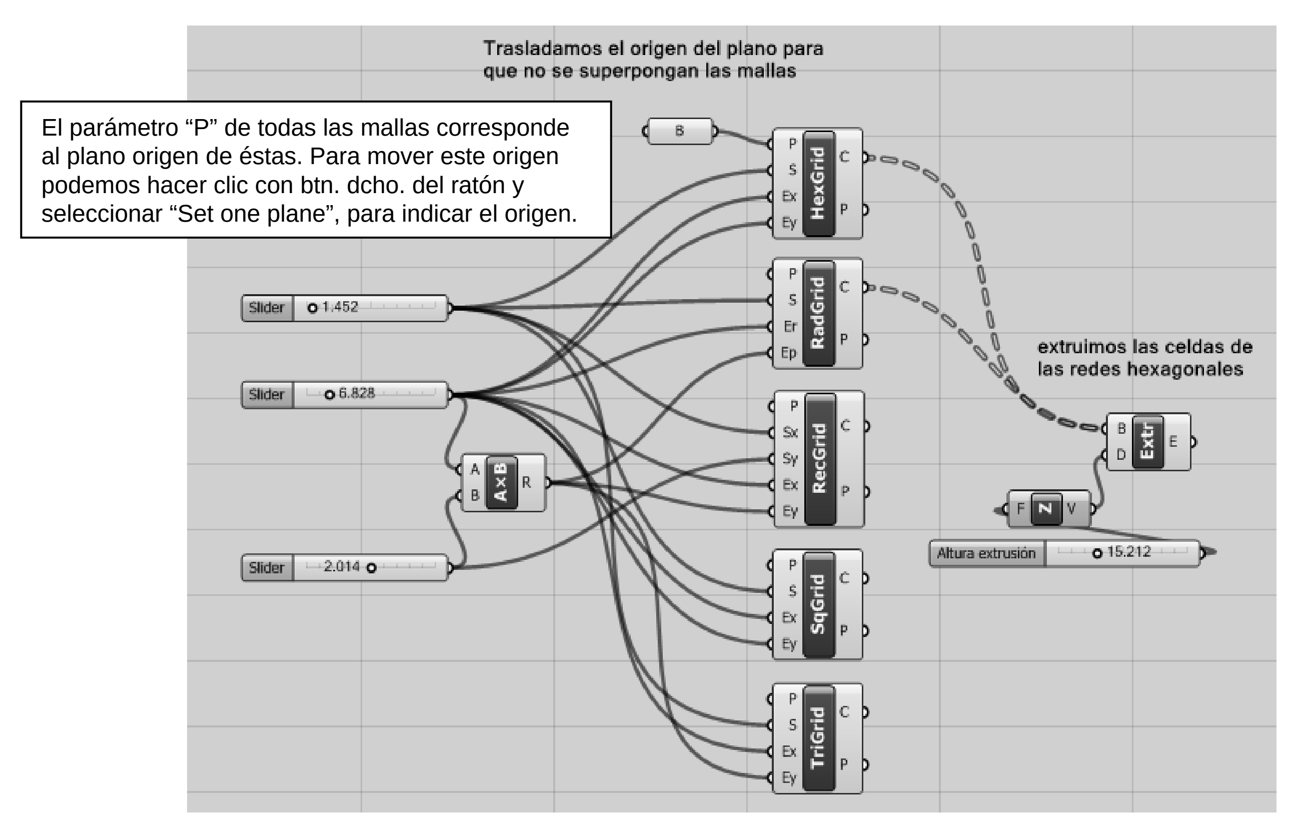
Y también podemos dibujar los puntos y construir la superficie a partir de ellos.

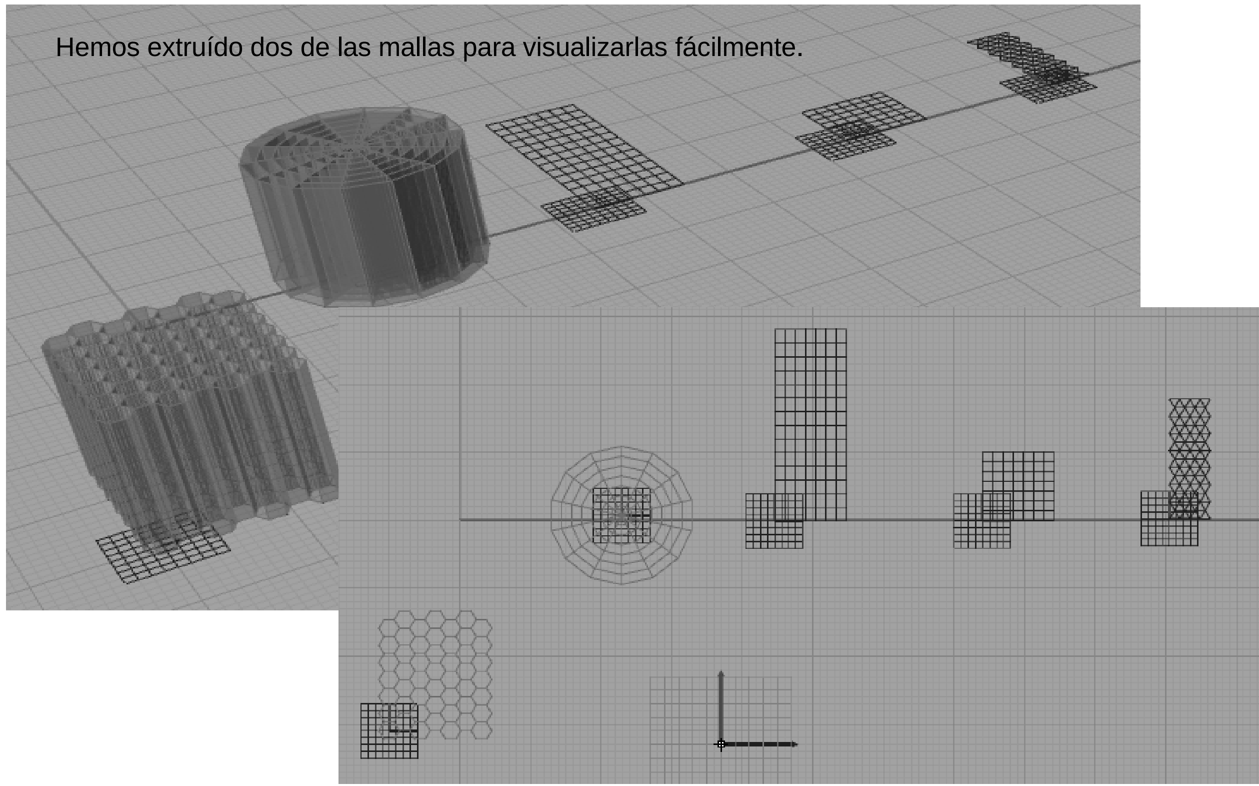




Mallas

Grasshopper nos permite dibujar mallas ("2D grid", en Vector/Grid): hexagonal, radial, rectangular, cuadrada y triangular. Incluimos una imagen en la que hemos creado una de cada tipo, compartiendo parámetros y desplazando el origen de todas ellas, para que no se superpongan unas con otras.





Índice

CUADERNO



Cuadernos.ijh@gmail.com
info@mairea-libros.com

